Polling systems and the power-series algorithm

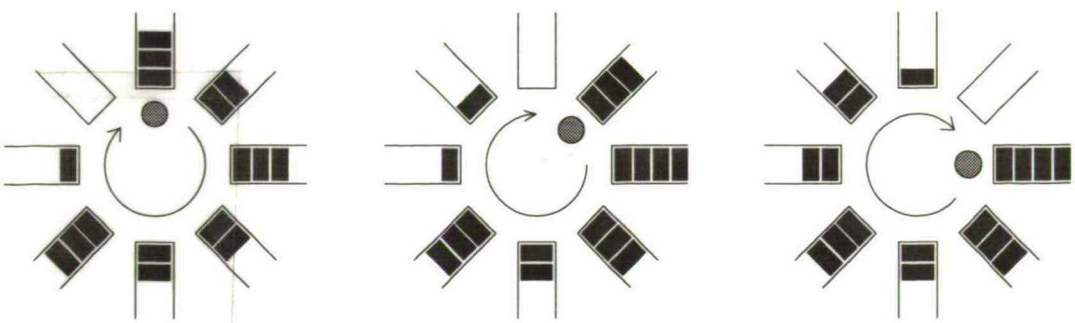
Mei, Robert Douwe van der

Publication date:
1995

Document Version
Publisher's PDF, also known as Version of record

Polling Systems and the Power-Series Algorithm

Rob van der Mei



Polling Systems and the Power-Series Algorithm

Polling Systems and the Power-Series Algorithm

Proefschrift

ter verkrijging van de graad van doctor aan de Katholieke Universiteit Brabant, op gezag van de rector magnificus, prof.dr. L.F.W. de Klerk, in het openbaar te verdedigen ten overstaan van een door het college van dekanen aangewezen commissie in de aula van de Universiteit op vrijdag 7 april 1995 om 14.15 uur

door

Robert Douwe van der Mei

geboren op 30 april 1966 te Tilburg

promotor : prof.dr.ir. O.J. Boxma
copromotor : dr. J.P.C. Blanc

Voorwoord

Veel dank ben ik verschuldigd aan begeleider en copromotor dr. J.P.C. Blanc, die met zijn deskundigheid een belangrijke bijdrage heeft geleverd in de totstandkoming van dit proefschrift. Hans, ik heb jouw begeleiding altijd bijzonder plezierig gevonden.
Daarnaast bedank ik promotor prof.dr.ir. O.J. Boxma voor zijn enthousiaste en professionele begeleiding. Onno, jouw ideeën, zowel over wachtrijanalyse als over sport, waren verfrissend.
Een bijzonder woord van dank gaat uit naar kamergenoot Marc van Eijs, die in belangrijke mate heeft bijgedragen tot de leuke tijd die ik op de K.U.B. heb gehad. Onze vele gesprekken en tafeltennispartijen zorgden voor de nodige ontspanning.
Behalve Marc wil ik Wilbert van den Hout, Loek Schoenmaker, Bart Smit en mijn vader hartelijk bedanken voor het kritisch lezen van delen van het manuscript.
Sem Borst bedank ik voor de prettige samenwerking die heeft geleid tot hoofdstuk 6 van dit proefschrift.
Ook ben ik dank verschuldigd aan Paul Smit voor zijn hulp bij het ontwerpen van de omslag van dit proefschrift en het maken van de plaatjes en grafieken.
I would like to thank Hanoch Levy for giving me the opportunity to visit Rutgers University. It was a pleasure to work with him.
Tenslotte, Tien, bedankt voor de steun die je me al die jaren hebt gegeven.

Rob van der Mei

Tilburg, februari 1995

Contents

Chapter 1

Introduction

1.1 Motivation

The phenomenon of queueing arises in many real-life situations. Waiting lines occur at post offices, in traffic situations and at elevators, but also on a more abstract level in communication systems and computer networks, in which information (voice, video, data) has to be transported from one place to another. The undesirability of congestion phenomena has raised the need to reach a better understanding of queueing situations. For this purpose, performance analysts develop queueing models, study their behavior and attempt to optimize their performance.
The main entities in queueing models are *customers* which arrive at a station requiring service from a *server*. Arriving customers to whom service can not be rendered immediately take place in a *queue*. Queueing models are typically of a stochastic nature in the sense that the service times and interarrival times of successively arriving customers are not exactly known a priori, but are assumed to have some (known) probability distribution. As a consequence, queueing models can not be expected to predict the exact delay of an arriving customer. Instead, queueing models aim to make probabilistic statements about the main quantities of interest. Typical performance measures are means, standard deviations and tail probabilities of waiting times or queue lengths.

The most extensively studied queueing models consist of a single queue served by a single server (cf. [65]). The results obtained for these seemingly simple models have contributed to the understanding of fundamental characteristics of queueing systems. However, in recent applications the distributed and parallel nature of service facilities has led to queueing models with more than one queue and more than one server.
This thesis is mainly devoted to the analysis of a class of multiple-queue models called *polling models*. Basically, a polling model consists of a number of queues, attended by a single server who visits the queues in some order to

render service to the customers waiting at the queues, typically incurring some switch-over time while moving from one queue to the next. Figure 1.1 shows a typical plot of a polling model. The term 'polling' originates from the so-

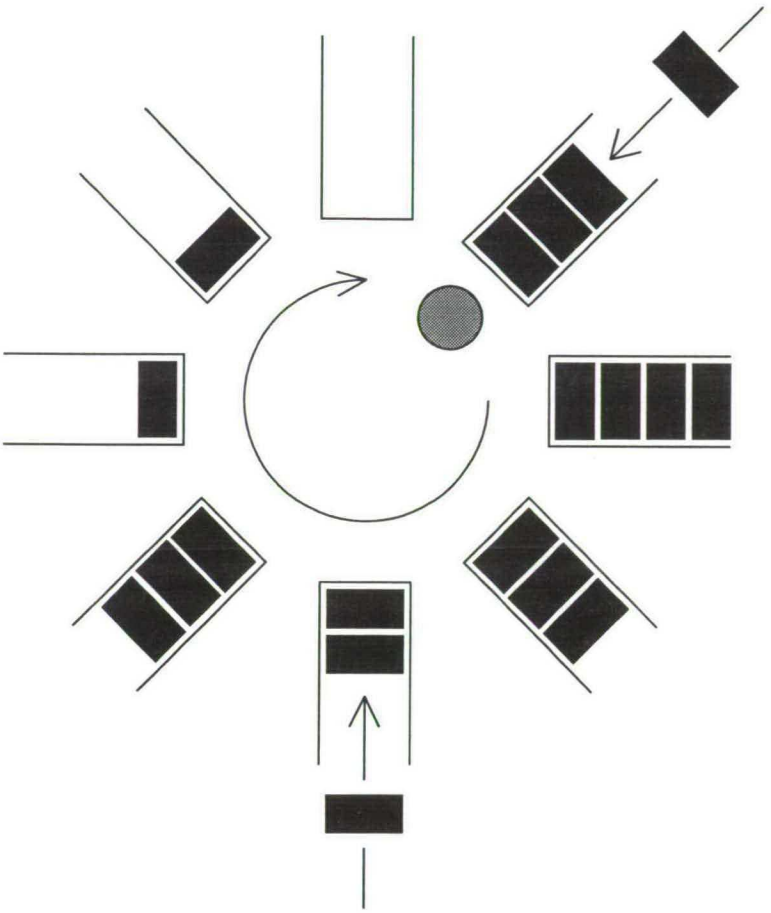


Figure 1.1: Polling model.

called polling data link control scheme, in which a central computer (server) interrogates each terminal (queue) on a communication line to find whether it has any information (customers) to transmit. The addressed terminal transmits information and the computer then switches to the next terminal to check whether that terminal has any information to transmit. In a broader perspective, polling models are applicable in situations in which several types of user compete for access to a common resource which is available to only one type of user at a time. Due to their broad applicability, many polling models have been studied in the literature during the last few decades (cf. [158], [159], [161]

for extensive overviews). However, only a limited class of polling models allows an exact analysis and even if polling models are exactly analyzable at all, ingenious algorithms may have to be applied to obtain numerical data for important performance measures, such as the mean waiting times at the queues. For this reason, several numerical algorithms have been developed to compute performance measures for polling models. In this thesis we direct most of our attention to the so-called power-series algorithm (PSA), a numerical tool for the analysis of a broad class of polling models.

The remainder of this chapter is organized as follows. Section 1.2 gives a global overview of various aspects of polling systems. Due to the diversity of polling models, we restrict ourselves in this thesis to the analysis of a number of specific polling models. The various models considered here are different, though they have some common features. To avoid overlap in the model descriptions, we give in section 1.3 a basic model description of the common characteristics of the various models considered. Finally, an overview of the contents of the thesis is given in section 1.4.

1.2 Polling systems

This section gives an overview of various aspects of polling systems. In section 1.2.1 we discuss the main applications of polling models. Section 1.2.2 contains a brief overview of the variety of polling models that has been studied in the literature. In section 1.2.3 we give a survey of the state-of-the-art in the analysis of polling models. Section 1.2.4 contains an overview of numerical algorithms that can be used for the analysis of polling models. In section 1.2.5 we discuss the main results that have been obtained in the optimization of polling models.

1.2.1 Applications

In this section we discuss some applications of polling models. For more detailed information on the applicability of polling models, the reader is referred to extensive surveys in Grillo [92], Takagi [160] and Levy and Sidi [121].

Computer-communication systems
Recent developments in the area of computer-communication systems have led to strong requirements on their efficiency and flexibility. To meet these requirements, modern communication systems mostly have a distributed and parallel structure. Many of these distributed systems can be modeled as polling models. An important class of systems that can be modeled as polling models is constituted by the class of so-called Local Area Networks (LANs). A LAN consists of a number of stations or computers interconnected by a common communication medium for transmitting packetized information. Recently, due to the integration of real-time traffic (voice, video) and non-real-time traffic (data), the access protocols for LANs, whose main target is to support non-real-time

traffic, are no longer viable. This development has given rise to the design of Metropolitan Area Networks (MANs), which aim to support real-time as well as non-real-time traffic (cf. [69]). Various variants of MANs can be modeled as polling models (cf. e.g. [5]). To avoid collisions when different stations want to transmit information at the same time, several conflict-free protocols have been designed.
One type of protocol used in communication networks is the so-called token ring, in which a token is circulated along the stations, representing the right for transmission. When a station receives the token, it may start transmitting packets. After departing from that station, the token proceeds to the next station. There are several token-ring protocols describing the behavior of the token. The reader is referred to Takagi [160] and Bux [57] for more detailed discussions about token-ring protocols. Another type of protocol for avoiding collisions between stations that want to transmit information simultaneously is the so-called slotted ring, in which the communication ring is divided into a number of time slots, circulating at a constant speed along the queues. When an empty slot passes by a station that wants to transmit a packet of information, the packet is put into the slot.
A token ring may be viewed as a polling system where the server represents the token and the queues represent the stations. Evidently, multiple-server polling models may be used to model token-ring networks in which the stations are interconnected by multiple token rings rather than by a single token ring. A slotted ring may generally be viewed as a multiple-server polling system, where each slot is represented by a server.

Maintenance, manufacturing, production
Polling models also find applications in the area of maintenance. For instance, a polling model can be used to describe a patrolling repairman who inspects a number of machines to check whether a breakdown has occurred and if so, eliminates such breakdowns (cf. [129]). Evidently, in polling models the repairman is represented by the server, the breakdowns are represented by the customers and the times needed by the repairman to travel from one machine to the next are represented by the switch-over times. In [56], [103] and [105] similar models are studied in which an operator at a fixed position serves a number of storage locations on a rotating carousel conveyor. Weststrate [171] studies a polling model which captures the behavior of a single repairman who is not only concerned with corrective maintenance (i.e. maintenance after a breakdown has occurred), but who can also perform preventive maintenance.
Polling models are also useful for modeling flexible manufacturing and production systems, where machines can be used to perform various types of task. Here, the server typically represents the machine, each of the queues represents a different type of job and the switch-over times represent the times needed by the machine to change from one type of operation to another. A similar application is multi-product economic lot scheduling, in which different types of product have to be processed by a single machine (cf. [144]).

Miscellaneous
Other applications of polling models are found, among others, in transportation networks (cf. [49]), public transportation systems (cf. [75]), vehicle-actuated signal control (cf. [138]), shipyard loading (cf. [74]), videotex (cf. [125]), mail delivery (cf. [135]) and elevators (cf. [89], [90]).

1.2.2 Polling models

Motivated by the variety of applications, a diversity of polling models has been studied during the last few decades. To give a structural overview of the variety of polling models considered in the literature, we successively discuss model variants with respect to the arrival process, the buffer size, the service process, the switch-over process, the server routing, the service discipline and the queueing discipline. This section aims to give the reader an idea of the variety of polling models that has been studied. For a more extensive overview the reader is referred to Takagi [158], [159], [161].

Arrival process
In polling models it is almost exclusively assumed that customers arrive at the queues according to mutually independent homogeneous single Poisson arrival processes. In many cases arrival processes can be described adequately by Poisson processes, e.g. in the case of telephone calls and traffic accidents. Moreover, due to the memorylessness property of the Poisson process, polling models with Poisson arrivals are relatively easy to analyze.
Yet, in many situations the assumption of Poisson arrivals is unrealistic, e.g. in case of bursty traffic (voice, video). To model arrival processes with dependency structures between the interarrival times, Lucantoni [128] has introduced the so-called Batch Markovian Arrival Process (BMAP). To the best of the author's knowledge, no papers have appeared about the analysis of polling models with general BMAPs.
In most polling models, customers arrive from some external infinite population. In some applications this assumption is unrealistic, e.g. in maintenance environments where customers represent machine breakdowns. Altman and Yechiali [8] study a *closed* polling model in which customers, after having received service at a queue, proceed to another (possibly the same) queue. Sidi and Levy [151] consider a system in which (external) customers arrive at a queue according to a Poisson process, and in which a customer, after having received service at a queue, either leaves the system or moves to another queue (with some given probability). Levy and Sidi [122] study a polling model with simultaneous arrivals at the queues.

Buffer size
In most polling models the buffer size is assumed to be infinite. In a number of applications the buffer capacities of the queues are obviously finite (e.g. transportation, manufacturing). In some applications it is natural to assume that the buffers are unit sized, i.e. each queue can only accommodate one customer

at a time. Applications of polling models with unit-sized buffers are found e.g. in the area of maintenance.

Service and switch-over process
The service times at a queue are typically assumed to be samples from a probability distribution which is characteristic for that queue. The service times are usually assumed to be mutually independent and independent of the actual state of the system.
The switch-over times needed by the server to move from one queue to another queue are typically assumed to be samples from some prespecified probability distribution which is characteristic for that couple of queues. In a few cases the switch-over times are assumed to be decomposed into switch-out times (which are characteristic for the queue being departured from) and switch-in times (which are characteristic for the queue being switched to), putting a special structure on the switch-over times.
The switch-over times are usually assumed to be mutually independent and independent of the current state of the system.

Server routing
The order in which the server visits the queues is determined by some routing mechanism. Such a mechanism may depend on the actual state of the system (dynamic) or may be independent of the state of the system (static).
We first discuss *static* routing mechanisms. The traditional routing mechanism is the cyclic server routing. To model systems in which particular queues are visited more frequently than others, cyclic polling has been extended to periodic polling, in which the server visits the queue *periodically* according to some service order table of finite length (cf. [110], [13]). Alternatively, the server may be routed along the queues according to some *probabilistic* routing mechanism. Kleinrock and Levy [104] introduce the so-called random polling mechanism in which, after a departure from a queue, the server proceeds to queue j with some given probability p_j. Boxma and Weststrate [48] generalize the random polling scheme to the so-called Markovian routing scheme in which the server, after a departure of the server from queue i, proceeds to queue j with given probability $p_{i,j}$.
Under *dynamic* server routing, the decision of the server as to the order in which the queues are visited may depend on a certain amount of information available to the server, such as the queue lengths. For instance, it might not make sense to move to an empty queue while customers are waiting at other queues. An example of dynamic server routing in systems with full information about the buffer contents is the 'serve-longest-queue' policy.
Recently, Yechiali [176] has introduced the so-called *semi-dynamic* server routing, in which the server, at the end of each tour along the queues, receives information about the queue lengths. Based on this information, the server makes a decision as to the order in which it will visit the queues in the next tour. This order cannot be changed during the course of that visit tour.

Service discipline
The service discipline specifies the number of customers that is served during one visit of the server to a queue. The most common service strategies are the *exhaustive* service discipline, under which the server continues to work until the queue has become empty, and the *gated* service discipline, under which exactly those customers are served who were present at the queue at the beginning of the visit.
In the literature, a whole abundance of service disciplines has been proposed by putting some cut-off mechanism (which eventually depends on the evolution of the queue length during the server visit) on the classical exhaustive and gated service disciplines. The service disciplines can be classified into the class of *customer-limited* service disciplines, in which restrictions are put on the number of customers served during a visit of the server to a queue, and the class of *time-limited* service strategies, putting restrictions on the amount of time spent by the server during one visit of the server to a queue. Alternatively, service disciplines can be classified into the so-called *exhaustive-type* policies and the *gated-type* policies, depending on whether customers who arrive at a queue while the server is working at that queue are *candidates* for service during the same visit of the server to that queue (cf. e.g. [123]). Under an exhaustive-type policy the customers arriving at a queue in service are candidates for service in the same visit period, whereas under a gated-type policy they are not. Numerous hybrid variants of service disciplines can be conceived by combining the various types of cut-off mechanisms. A number of these service discipline has been studied in the literature, such as customer-limited type service policies like the k-limited service, Bernoulli service (cf. [102]), probabilistically-limited service (cf. [114]), binomial-gated service (cf. [118]), fractional-exhaustive service (cf. [117]), Bernoulli-type service (cf. [142]), and time-limited service disciplines with exponential time limits (cf. [64], [115]) and with constant time limits (cf. [73]).

Queueing discipline
The queueing discipline specifies the *order* in which the customers present at the same queue are served. The most common queueing discipline is the classical First-Come-First-Served (FCFS) discipline. It is important to note that the queue-length distribution is independent of the service order, and so are, by Little's law, the mean waiting times (provided the queueing discipline does not depend on the service times). However, the distribution of the waiting time *does* depend on the queueing discipline.

1.2.3 Analysis

Performance analysis of polling models is important for the prediction and understanding of the performance of the systems described by these models. Common system performance measures are the (joint or marginal) probability distributions of

(1) the number of customers present at each of the queues;

(2) the waiting time of an arriving customer;

(3) the total amount of work in the system, i.e. the total amount of time the server would need to empty the system if no new arrivals would occur (switch-over times excluded);

(4) the cycle time, i.e. the time interval between two successive visits (departures) of the server to (from) a particular queue.

The time-dependent behavior of polling models is generally very hard to analyze. To the best of the author's knowledge, hardly any detailed exact results for polling models under the transient regime have been obtained. Recently, Choudhury and Whitt [62] have developed a numerical technique that allows for the numerical computation of time-dependent as well as time-independent performance measures of a class of polling models (cf. also section 1.2.4).
The vast majority of the available literature on the analysis polling models is devoted to the *steady-state* (long-run) behavior of the system. To this end, one typically assumes that the system is stable, and describes the process as a (continuous- or discrete-time) Markov chain for which the existing ergodic distribution is determined by a set of balance equations. The reader is referred to Fricker and Jaïbi [85], [86] for rigorous proofs of necessary and sufficient conditions for the *stability* of static polling strategies. For dynamic polling strategies there are hardly any exact results known for stability.
In virtually all polling studies that have been published in the literature, it is assumed that the customers arrive according to a homogeneous single *Poisson arrival process*. Throughout this chapter, this assumption is supposed to be satisfied, unless indicated otherwise.
In the remainder of this section we give an overview of the main exact results that have been obtained in the analysis of polling models. For more detailed discussions, we refer to Takagi [158], [159], [161].

The variety of polling studies that has been published in the last few decades has revealed a striking difference in complexity. On the one hand, polling models with the exhaustive and the gated service disciplines allow an exact detailed analysis. On the other hand, models with limited-like service disciplines can only be solved in special cases. Recently, this sharp distinction has been illuminated by Resing [142] and Fuhrmann [87], who independently identify a class of service disciplines which are particularly easy to analyze. This class contains the service disciplines which satisfy the following property.

Additivity Property
If the server arrives at a queue to find a number of customers there, then at the end of the server's visit, each of these customers has been effectively replaced by an independently identically distributed (i.i.d.) population of customers.

For instance, the gated service discipline satisfies the Additivity Property, because at the end of the server's visit, each customer C present at the beginning of the visit (commonly referred to as a *polling instant*) has been effectively replaced in an i.i.d. manner by all customers who have arrived during the service of C. Similarly, it is readily verified that the exhaustive service discipline also satisfies the Additivity Property. Yet, the 1-limited service discipline for instance does *not*. To see this, note that at the end of the server's visit, the first served customer C present at the polling instant at that queue has been effectively replaced by a population P_0 of all customers who have arrived during the service of C, whereas the other customers present at the beginning are not served at all, so that each of them is virtually replaced by a population P_1 consisting of a single customer at the queue under consideration. Evidently, P_0 and P_1 do not have the same probability distribution.

For *cyclic* (*periodic*) polling models in which the service discipline at each of the queues satisfies the Additivity Property, Resing [142] shows that the joint queue-length process at embedded polling instants *at a given queue* constitutes a Multi-Type Branching Process (MTBP) with immigration in each state (cf. e.g. [10]). Based on this observation, he uses the theory of MTBPs to obtain exact expressions for the joint queue-length distribution at these embedded epochs.
It should be noted that polling models in which each of the service disciplines satisfies the Additivity Property, and in which the server routing is *probabilistic* (instead of periodic), can *not* be included in the framework of MTBPs (cf. [142]).
In addition to the exact characterization of the MTBP-models, recent developments have shown that simple relations can be given between polling models with and without switch-over times (cf. [87], [72], [155], [33]).
Although the class of MTBP-models allows, at least formally, an exact analysis, the problem of efficiently determining numerical values for performance measures like mean waiting times and mean queue lengths is generally non-trivial. We refer to section 1.2.4 for a discussion of numerical techniques to compute performance measures for such polling models.

For polling models which are *not* of MTBP-type, detailed exact results are very scarce and mainly restricted to two-queue models. For models in which one of the two queues is served exhaustively, the waiting-time distribution at polling instants has been derived for the case the other queue is served according to the 1-limited [94], Bernoulli [172], and k-limited service policy [113]. For two-queue models in which none of the queues is served exhaustively, the problem of determining the joint queue-length distribution at polling instants can in some cases be translated into boundary-value problems for complex functions (Riemann, Riemann-Hilbert, Wiener-Hopf, Dirichlet) and singular integral equations, requiring ingenious techniques to obtain numerical values of system performance measures (cf. [68] for a detailed discussion on the technique of boundary-value problems and [37] for its applications to two-queue polling

models). In this way, various two-queue polling models have been solved, such as the 1-limited/1-limited case (cf. [78], [68], [41]), the Bernoulli/Bernoulli case (cf. [112]) and the two-queue case with (exponentially) time-limited service (cf. [64]).
For polling models which are not contained in the framework of MTBPs and which have more than two queues, exact detailed analysis (e.g. of the waiting-time and queue-length distributions) is extremely complicated.

Considerable progress in the analysis of polling models has been made by Boxma and Groenendijk [40], who have extended the fundamental property of *work conservation* for polling models with zero switch-over times to the *work decomposition* principle for polling models with non-zero switch-over times. This observation has led to a general framework for deriving *pseudo-conservation laws* (PCLs), i.e. exact expressions for a specific weighted sum of the mean waiting times at the queues, in polling models with general mixtures of service disciplines (cf. also [38], [94]). Many extensions of the PCL presented in [40] (for cyclic polling with exhaustive, gated and 1-limited service) have been derived for a number of models, such as polling models with periodic polling (cf. [43]), probabilistic server routing (cf. [48]), compound Poisson arrival processes (cf. [42]) and multiple-priority classes (cf. [84], [150]), and also for discrete-time models polling (cf. [42]) and polling models with a dormant server (cf. [30]).
The discovery of the PCLs has not only contributed to the understanding of the behavior of polling systems, but has also given rise to the construction of a variety of waiting-time approximations for polling models with general mixtures of service disciplines. Groenendijk [93] presents a general approach for deriving PCL-based waiting-time approximations for models with mixtures of exhaustive, gated and 1-limited service. This approach has been applied to a number of other service disciplines, such as the Bernoulli service discipline (cf. [163]) and the k-limited service strategy (cf. [59], [88]).

1.2.4 Numerical techniques

In this section we give an overview of available numerical techniques to analyze the performance of multiple-queue models, which can be applied for the analysis of polling models.

Efficient techniques for polling models with an MTBP-structure
We will now discuss a number of available techniques to compute performance measures for the class of polling models that allow an MTBP-interpretation (cf. section 1.2.3). The complexity of these algorithms is commonly indicated in terms of the number of *elementary operations*, like additions and multiplications. Throughout, s will stand for the number of queues, ρ will be the offered load to the system, and ϵ will indicate the required accuracy of the computations.

The Buffer Occupancy Technique (BOT) considers so-called buffer-occupancy variables, $B_{i,j}$, indicating the length of queue j at polling instants at queue i. To obtain the time-average mean waiting times at the queues, the unknowns $EB_{i,i}^2$ are needed which, in turn, requires the solution of a set of s^3 linear equations with unknowns $EB_{i,j}B_{i,k}$. This set of equations can be solved iteratively, requiring $O(s^3\ log_\rho \epsilon)$ elementary operations. The BOT has been applied to almost all variations of polling models with an MTBP-structure (cf. e.g. [70], [71], [77]).

Another method for solving the mean waiting times is the Station Time Technique (STT), introduced in [83]. The STT considers station-time variables, S_i, indicating the length of a visit of the server to queue i plus the preceding/following switch-over time. The time-average mean waiting times can be expressed in terms of the unknown quantities ES_iS_j, and a set of $O(s^2)$ linear equations with variables ES_iS_j can be determined. The latter set of equations can be solved iteratively, requiring $O(s^2\ log_\rho \epsilon)$ elementary operations. The STT is applicable to cyclic (periodic) polling models with either exhaustive or gated service at all queues.
Sarkar and Zangwill [145] refine the STT by expressing the s^2 unknown quantities ES_iS_j in s unknowns ES_i^2, deriving a set of only s linear equations for the unknowns ES_i^2. However, the so-obtained set of linear equations is non-sparse, typically requiring $O(s^3)$ elementary operations by using standard methods for solving sets of linear equations.

Konheim et al. [107] introduce the Descendant Set Technique (DST) for computing moments of the waiting times. The DST considers buffer-occupancy variables at successive polling instants at a fixed queue. The key observation is that each customer who is present in the system at an arbitrary embedded epoch, is at the next embedded epoch (i.e. the beginning of the next cycle) effectively replaced in an i.i.d. manner by a family of customers. Based on this observation a simple iterative scheme can be derived to determine the moments of the buffer-occupancy variables at the embedded epochs, requiring only $O(s\ log_\rho \epsilon)$ elementary operations to obtain the moments of queue lengths at a single queue. From these quantities the moments of the time-average mean waiting times can be directly obtained. The DST is a flexible method that is readily applicable to systems with mixed service strategies, non-cyclic periodic server routing, customer routing, correlated arrivals and to discrete-time (slotted) models. The main disadvantage of the DST is that its efficiency degrades significantly when the system is heavily loaded.
To overcome this difficulty, Srinivasan et al. [154] present the Individual Station Technique (IST). To obtain the mean waiting time at a single queue (with numerical error $O(1/L!)$), a set of L linear equations has to be solved, and it requires $O(s^2)$ elementary operations to obtain the coefficients of these equations. The IST is not iterative, as opposed to the DST, and the accuracy typically does not degrade significantly when the system is heavily loaded. Based on the different characteristics, the DST and the IST can be considered as comple-

mentary to each other.

Based on the above-mentioned efficient techniques for computing the moments of the waiting times and the queue lengths, Federgruen and Katalan [82] propose an efficient approximation method to obtain the steady-state *distributions* of the queue lengths and the waiting times in models with exhaustive or gated service at each queue. Their approach requires $O(\max(s, K^2))$ elementary operations to compute the first K probability density function values of the joint queue-length distribution.

Recently, Choudhury and Whitt [62] have proposed the Numerical Transform Inversion Technique (NTIT). The NTIT is based on an efficient recursive algorithm for computing transform values in combination with a numerical inversion algorithm to compute distributions and moments of performance measures of the system. This NTIT requires only $O(s^{\alpha}\ log_{\rho}\epsilon)$ elementary operations to compute the moments of the waiting times at a single queue, where α is typically between 0.6 and 0.8. The NTIT, which is applicable to all MTBP-type polling models, can also be used to determine time-dependent performance measures of the system.

The above-discussed techniques make use of the MTBP-structure of the models considered here, and are therefore not applicable to polling models which do not allow an MTBP-interpretation. Those models are intrinsically more complicated and the numerical techniques to compute performance measures are computationally much more involved. We will now discuss a few numerical techniques for the analysis of these complicated models.

The power-series algorithm
The power-series algorithm (PSA) is a device for solving the steady-state distribution of a class of multiple-queue models. The PSA typically requires a Markov representation of the form $\{(\boldsymbol{N}(t), \boldsymbol{\Phi}(t)), t \geq 0\}$ on the state space $\mathbb{N}^s \times \mathcal{S}$, where $\boldsymbol{N}(t)$ stands for the joint queue-length vector and $\boldsymbol{\Phi}(t)$ for a supplementary variable (which may be used, e.g. to model the non-exponentiality of service or interarrival times) at time t, $t \geq 0$. The supplementary space $\mathcal{S}$ is typically assumed to be finite. In addition, it is assumed that the Markov process has a multi-dimensional quasi birth-and-death (QBD) structure. That is, transitions from state $(\boldsymbol{n}, \varphi)$ are only possible to states $(\boldsymbol{n}, \psi)$, $(\boldsymbol{n} + \boldsymbol{e}_j, \psi)$, and $(\boldsymbol{n} - \boldsymbol{e}_j, \psi)$, where $\boldsymbol{e}_j$ stands for the j-th unit vector, $j = 1, \ldots, s$.
Under the assumption that the process is irreducible and ergodic, there exist steady-state probabilities $p(\boldsymbol{n}, \varphi)$, $(\boldsymbol{n}, \varphi) \in \mathbb{N}^s \times \mathcal{S}$, which can in principle be obtained by solving the set of global balance equations of the system. Yet, the set of balance equations may be large or even infinite, and hence difficult to solve. The basic idea of the PSA is to *transform* the *non-recursively* solvable set of global balance equations into a *recursively* solvable set of equations by adding one dimension to the state space. This transformation is realized by expressing the state probabilities as power series in some formal variable χ at

the origin (cf. [167], [97]): for $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, \boldsymbol{\varphi}) = \sum_{k=0}^{\infty} b(k; \boldsymbol{n}, \boldsymbol{\varphi}) \chi^k. \tag{1.1}$$

Under some weak assumptions it can be shown (cf. [166]) that the first $n_1 + \cdots + n_s$ coefficients of the power series (1.1) vanish. Based on this property, a recursive computational scheme to calculate the coefficients of these power series can be obtained by substituting these power-series expansions into the global balance equations and equating corresponding powers of χ.

The basic idea of the power-series expansions stems from Hooghiemstra et al. [97]. The algorithm has been further developed by Blanc, which has led to more efficient implementations of the algorithm. The PSA has been applied to several types of multiple-queue models with single arrivals, such as the shortest-queue model (cf. [15], [23]), the coupled-processor model (cf. [17], [97]), and a variety of polling models (cf. [5], [6], [18]-[22], [27], [169], [170]). Van den Hout and Blanc [166] have extended the PSA to incorporate BMAPs for single-queue models, and have generalized this extension to the class of Markovian queueing networks with Multi-queue Markovian Arrival Processes (MMAPs), which can be viewed as multiple-queue generalizations of the BMAP (cf. [167]). Recently, Koole [109] has shown that the PSA is, formally, applicable to general Markov processes. He also states that the PSA is applicable to *discrete-time* Markov processes as well.
There are two main complications in the application of the PSA, being the convergence of the power series and the required amount of memory to store the coefficients of the power series. The reader is referred to chapter 2 for an extensive discussion about the PSA.

Leung's algorithm based on Discrete Fourier Transforms
Leung [114] presents a numerical technique based on Discrete Fourier Transforms (DFTs) to solve the joint queue-length distribution in cyclic polling models in which the queues are served according to a so-called probabilistically-limited service strategy, i.e. in which the maximal number of customers served during a visit of the server to a particular queue is a (discrete) random variable. To this end, he considers the embedded Markov chain formed at successive visit-completion instants, and derives a functional equation for the probability-generating functions (PGFs) of the joint queue-length at these server-departure epochs. To solve this functional equation, the equation is converted into a set of relations in terms of DFTs (after truncation of the state space). A fixed-point iteration is performed to find the DFTs of the embedded process. Based on these DFTs, one may obtain the PGFs of the marginal waiting-time distributions and the LSTs of the queue-length distribution at the various queues at server-departure epochs. From these distributions, the time-average waiting-time and queue-length distributions can be obtained. The algorithm is also applied in [115] to cyclic polling models with non-preemptive time-limited ser-

vice with exponential time limits. For this algorithm, a formal proof of the convergence of the iterations has not yet been obtained. Similar to the PSA, the order of magnitude of memory and CPU time required by the algorithm are intrinsically exponential in the number of queues, so that the use of the algorithm is restricted to rather small models.

Simulation
Simulation is a widely used technique for computing performance measures of all kinds of models, such as queueing models (cf. e.g. [132], [146]). However, in spite of their enormous flexibility, simulation techniques may be rather inefficient in many cases. For instance, when in a polling model the switch-over times are small, the majority of (discrete) events will be switch-over completion epochs. This is because the server will be quickly spinning around in the system when the system is empty for some time interval. Moreover, in many cases the results based on simulation are relatively inaccurate compared with numerical algorithms such as the PSA (with given bounds on the computation time). Blanc [21] makes a comparison of the performance of the PSA and Monte Carlo simulation. For a broad class of polling models exact expressions are known for some specific weighted sum of the mean waiting times at the queues (cf. section 1.2.3). Blanc compares the computed values of this quantity via the PSA and via simulation with the exact value. The results indicate that for small and moderately-sized systems the PSA performs significantly better than simulation.
The memory requirements of the PSA restrict the maximal number of terms that can be computed and hence, the accuracy of the computations. As a consequence, for large systems the computations with the PSA may become inaccurate. Hence, simulation, which is generally less memory consuming, may lead to better results for large systems. Furthermore, recent developments in the area of simulation techniques (score function method, importance sampling) allow estimation of the system performance of a *set* of models with only *one* simulation run, which has strongly improved the performance of simulation techniques. In addition, derivatives of the system performance measures can be obtained in one simulation run, opening many possibilities for optimization purposes (cf. e.g. [143] for an overview).

1.2.5 Optimization

In this section we discuss the main results that have been obtained for optimization of polling models. The interested reader is referred to Boxma [39] and Yechiali [176] for extensive surveys on static and semi-dynamic optimization, respectively.

The ultimate goal of system modeling and analysis is to obtain the 'best' possible system performance. To this end, one has to specify some performance measures to indicate what is meant by 'best'. In addition, the class of feasible system designs has to be specified, indicating the variety of models to choose

the 'best' from.
In general, there is a trade-off between 'efficiency' and 'fairness'. For instance, the exhaustive service discipline is generally considered to be efficient, but highly unfair, because a heavily-loaded queue may dominate the system. On the other hand, simply serving customers in the order of arrival would be ultimately fair, but may cause instability of the system. In view of this trade-off, general performance measures are commonly taken to be

$$\sum_{i=1}^{s} \rho_i \mathrm{E}W_i, \tag{1.2}$$

the mean amount of waiting work in the system or, more generally,

$$\sum_{i=1}^{s} c_i \mathrm{E}W_i, \tag{1.3}$$

a weighted sum of the mean waiting times at the queues, where the (non-negative) weights c_i represent the relative importance of the queues.
The class of feasible system designs is commonly restricted by technical constraints, such as the available information about the queue lengths and the structure of the network. For instance, if the server does not have any information about the queue lengths, the 'serve longest queue'-policy is infeasible. Alternatively, in a system with a ring topology non-cyclic server routing may be infeasible. Depending on assumed technical constraints, optimization problems are commonly referred to as static, semi-dynamic and dynamic.

Static optimization
Boxma et al. [45] consider polling models with periodic server routing and study the problem of determining a polling order table which minimizes (1.2). They suggest a simple approach which is based on the well-known Golden Ratio procedure (cf. [98]) for spacing the visits within the polling table. In [44] the same authors generalize these results to the more general problem of finding a service order table which minimizes (1.3). Kruskal [110] considers a similar problem in the case of deterministic interarrival, service and switch-over times. Blanc and Van der Mei [17] consider optimization of cyclic polling models with respect to the service disciplines. For cyclic polling models with Bernoulli service disciplines at the queues, they address the problem of determining a combination $(q_1, \ldots, q_s)$ of the Bernoulli parameters which minimizes (1.3). The results of this study are reported in chapter 3 of this thesis. Borst et al. [35] consider the similar problem of determining service limits in cyclic polling models with fixed service limits $(k_1, \ldots, k_s)$. For the problem of minimizing (1.3) they propose simple approximations.
Borst et al. [34] consider a system operated with a fixed-time polling (FTP) scheme, i.e. a scheme in which not only the visit order but also the starting times of the visits are specified. They discuss the problem of determining an FTP scheme which minimizes (1.3), and propose a heuristic approach based on successive determination of the visit frequencies, the visit orders and the server

intervisit times.

Semi-dynamic optimization
Browne and Yechiali [53] consider the problem of finding the visit order, at the beginning of each cycle, that minimizes (maximizes) the mean duration of that cycle. For models with exhaustive and gated service and in which the switch-over times typically consist of a switch-out time to depart from a queue and a switch-in time to move to the next queue, simple index rules for the optimal visit order are obtained. Fabian and Levy [81] suggest that for symmetrical models the semi-dynamic service order which maximizes (minimizes) the expected cycle-time duration minimizes (maximizes) the mean waiting times.

Dynamic optimization
Liu et al. [125] consider the problem of identifying polling strategies that (stochastically) minimize the amount of unfinished work in the system at all time. They show that the server should serve each queue exhaustively. In addition, their results imply that for fully symmetrical models the server should remain at the last visited queue when the system is entirely empty and that the cyclic server routing is optimal in case the only information available is the set of previous decisions.
For models with zero switch-over times, in which the queue lengths are known and in which the server is allowed to choose the next queue to be visited after each service completion, the policy that minimizes (1.3) is given by the classical $c\mu$-rule (cf. [130], [58]). For systems with non-negligible switch-over times the situation is much more complicated, and complete solutions are restricted to two-queue cases (cf. [96], [108]).

1.3 Basic model description

In the remainder of this thesis we investigate several specific polling models, which have some characteristics in common. To avoid unnecessary overlap, we give a general model description in this section which is valid for each of the models studied in subsequent chapters. For details which are specific for the various models considered, we refer to the corresponding chapters.

The basic polling model considered throughout consists of s queues, $Q_1, \ldots, Q_s$, attended by a single server. All queues are assumed to have an infinite buffer capacity. Customers arrive at Q_i according to a Poisson arrival process with rate λ_i, $i = 1, \ldots, s$. The service times at Q_i are assumed to have a Coxian distribution with parameters Ψ_i^1, $\pi_i^{1,\xi}$, $\mu_i^{1,\xi}$, $\xi = 1, \ldots, \Psi_i^1$; that is, with probability $\pi_i^{1,\xi}$ a service is composed of subsequent phases ξ, $\xi - 1, \ldots, 1$, and the transition rate from phase ξ is $\mu_i^{1,\xi}$, $\xi = 1, \ldots, \Psi_i^1$, $i = 1, \ldots, s$. Consequently, the LST of the service-time distribution at Q_i is given by: for $i = 1, \ldots, s$,

$$\beta_i(\omega) = \sum_{\xi=1}^{\Psi_i^1} \pi_i^{1,\xi} \prod_{\psi=1}^{\xi} \frac{\mu_i^{1,\psi}}{\mu_i^{1,\psi} + \omega}, \quad \text{Re}\,\omega \geq 0. \tag{1.4}$$

Let $\boldsymbol{\beta}^{(k)} = (\beta_1^{(k)}, \ldots, \beta_s^{(k)})$ denote the vector of k-th moments of the service times at the various queues, $k = 1, 2$. The k-th moment of an arbitrary service time is given by: for $k = 1, 2$,

$$\beta_k := \frac{1}{\Lambda} \sum_{i=1}^{s} \lambda_i \beta_i^{(k)}, \text{ with } \Lambda := \sum_{i=1}^{s} \lambda_i. \tag{1.5}$$

Denote the offered load to Q_i and the total offered load to the system by:

$$\rho_i := \lambda_i \beta_i^{(1)}, \ i = 1, \ldots, s, \quad \rho := \sum_{i=1}^{s} \rho_i. \tag{1.6}$$

At each queue the queueing discipline is assumed to be FCFS. This assumption mainly serves the ease of interpretation, because we restrict ourselves to the analysis of *mean* waiting times, which are known to be insensitive to the order in which the customers are served (cf. section 1.2.2).
The service discipline at Q_i is taken to be the Bernoulli discipline with parameter q_i, $i = 1, \ldots, s$, which works as follows. When the server arrives at Q_i finding that queue non-empty, at least one customer at that queue is served; otherwise, the server proceeds to the next queue. Moreover, if after the completion of the service of a customer at Q_i there are still customers waiting at that queue, with probability q_i another customer at Q_i is served; otherwise, the server departs from Q_i to move to the next queue. A vector of Bernoulli parameters $\boldsymbol{q} = (q_1, \ldots, q_s)$ is referred to as a *Bernoulli schedule*. Note that the cases $q_i = 0$ and $q_i = 1$ correspond to the classical 1-limited and the exhaustive service discipline, respectively.
The switch-over times needed by the server to move from Q_i to Q_j are assumed to be Coxian distributed with parameters $\Psi_{i,j}^0$, $\pi_{i,j}^{0,\xi}$, $\mu_{i,j}^{0,\xi}$, $\xi = 1, \ldots, \Psi_{i,j}^0$, $i, j = 1, \ldots, s$; that is, with probability $\pi_{i,j}^{0,\xi}$ a switch-over time from Q_i to Q_j is composed of subsequent phases ξ, $\xi - 1, \ldots, 1$, and the transition rate from phase ξ is $\mu_{i,j}^{0,\xi}$, $\xi = 1, \ldots, \Psi_{i,j}^0$, $i, j = 1, \ldots, s$, so that the LST of the switch-over time distribution from Q_i to Q_j is given by: for $i, j = 1, \ldots, s$,

$$\sigma_{i,j}(\omega) = \sum_{\xi=1}^{\Psi_{i,j}^0} \pi_{i,j}^{0,\xi} \prod_{\psi=1}^{\xi} \frac{\mu_{i,j}^{0,\psi}}{\mu_{i,j}^{0,\psi} + \omega}, \quad \text{Re}\,\omega \geq 0. \tag{1.7}$$

Denote by $\sigma_{i,j}^{(k)}$ the k-th moment of a switch-over time from Q_i to Q_j, $i, j = 1, \ldots, s$, $k = 1, 2$.
Throughout, it is assumed that the interarrival times, the service times and the switch-over times are mutually independent and independent of the actual state of the system.

1.4 Overview of the thesis

In this section we give an overview of the contents of the remainder of this thesis. The results are based on a number of papers. The references of these papers are given. Finally, some notational conventions are introduced.

Chapter 2 basically consists of two parts. In the first part we give a survey of the use of the PSA for general multiple-queue systems for which the underlying process has the structure of a multi-dimensional QBD process. A general model description is given and conditions for applicability of the PSA are discussed. Moreover, we discuss a few techniques to improve the convergence of the power series. In addition, we give some general ideas for efficient implementation, which have led to strong improvements of the performance of the algorithm. In the second part of the chapter the use of the PSA is extended to the computation of *derivatives* of performance measures with respect to a general class of continuous system parameters. This extension is very useful for solving a variety of *optimization* problems and for analyzing the sensitivity of the system performance with respect to the system parameters. Subsequently, the complexity of the extension of the PSA is discussed and some notes on the practical implementation are given.
The extension of the PSA to the computation of derivatives presented in chapter 2 is used in the remaining chapters, in which we analyze and optimize the performance of a number of polling models by means of the PSA.

Chapter 3 considers *optimization* of polling models with respect to the service disciplines at the queues. We study cyclic polling models in which the queues are served according to a *Bernoulli schedule* $\boldsymbol{q} = (q_1, \ldots, q_s)$. It is our aim to find a Bernoulli schedule which minimizes (1.3), an arbitrary weighted sum of the steady-state mean waiting times at the various queues. However, the Bernoulli service discipline does not satisfy the Additivity Property, so that the model does not allow an MTBP-interpretation (cf. section 1.2.3). The present model generally not exactly analyzable. Moreover, the optimization problem considered here is not exactly solvable. As a demonstration of the general approach discussed in chapter 2, we show how the model can be analyzed by means of the PSA, with the extension to the computation of derivatives with respect to the Bernoulli parameters. This extension is then used to gain insight into the character of optimal Bernoulli schedules. Light-traffic asymptotes of the optimal Bernoulli schedule are obtained by algebraically determining the first few terms of the power-series expansions of the mean waiting times at the queues. Heavy-traffic asymptotes are found on the basis of the stability condition for the system. In addition, a (partly conjectured) partial solution to the optimization problem is given. This solution states that each Q_i for which the ratio c_i/ρ_i is maximal over all queues should be served exhaustively, i.e. $q_i = 1$. However, the time requirements of an optimization procedure based on the use of the PSA may be considerable. Therefore, we propose and test a simple and fast-to-evaluate approximation method to find nearly-optimal Bernoulli sched-

ules.

In chapter 4 we investigate the performance of polling models with *Markovian server routing*, i.e. in which the server is routed along the queues according to some discrete-time Markov chain. We demonstrate how the performance of the model can be analyzed by means of the PSA. We analyze the behavior of the model and compare its performance with the performance of a similar model under periodic server routing. Numerical results suggest that the mean total amount of unfinished work in the system is generally smaller in the case of periodic polling, but that this dominance relation is not generally valid for the individual mean waiting times. We consider the problem of optimizing the system performance with respect to the routing probabilities. Numerical experiments indicate the tendency of the optimal Markovian server routing towards deterministic routing decisions. We also observe that the various optimal routing matrices are of a special and easily interpretable structure. These observations contribute to the understanding of the behavior of polling systems with Markovian server routing. Based on these observations, we give some guidelines for the construction of optimal routing matrices.

In chapter 5 we consider a polling model with a so-called *dormant server*, i.e. a server which is allowed to rest at a queue, instead of having to move around, when the system is empty. This model is not contained in the general setting of chapter 2. We show how the PSA can be adapted to compute performance measures of the system. We investigate the improvements that can be made by allowing the server to rest at a queue when the system is empty. Numerical examples will show that the system performance can be strongly improved by allowing the server to rest at a queue, especially in lightly- or medium-loaded systems in which the switch-over times are considerable. In general, it is not optimal to allow the server to rest at the queue that is being visited just before the system becomes empty. We consider the problem of determining a set of queues at which the server should be allowed to rest when the system is empty, so as to minimize an arbitrary weighted sum of the steady-state mean waiting times at the various queues (cf. (1.3)). Because the problem is generally too involved to give explicit solutions, we propose and test a simple and fast-to-evaluate approximation.

In chapter 6 we investigate the performance of polling models with *multiple servers* in which each of the servers visits the queues independently in some fixed order. These models turn out to defy an analytic approach completely. We show how the PSA may be used to determine the joint distribution of the queue lengths and the positions of the servers in the system. From this distribution other relevant performance measures like mean waiting times of customers and utilization factors of the individual servers may be obtained. Numerical experiments are performed to investigate the tendency of the servers to cluster, to study the possibility of segmentation of the system into a number of subsystems and to make some comparisons with single-server models

with comparable load. Finally, we propose a new method to approximate the mean waiting times in multiple-server polling models with independent servers.

Publications
The results to be presented are based on a number of papers that either have appeared (or will appear) in the literature or have been submitted for publication. The results in chapter 2 are based on Blanc and Van der Mei [26]. In chapter 3 we present the results reported in Blanc and Van der Mei [25], [28]. Chapter 4 is a modified version of Van der Mei [169]. The results of chapter 5 have been reported in Blanc and Van der Mei [27]. Finally, chapter 6 is based on Van der Mei and Borst [170] and Borst and Van der Mei [36].

Conventions
Throughout this thesis, vectors and matrices are printed in bold face type. Vectors are printed in italic and matrices are printed in roman style. Sets are printed in calligraphic style. A vector $\boldsymbol{v}$ in an s-dimensional vector space $\mathcal{V}^s$ consists of components $(v_1, \ldots, v_s)$. For $\boldsymbol{v} \in \mathcal{V}^s$ we define $|\boldsymbol{v}| := v_1 + \cdots + v_s$. The vector $\boldsymbol{e}_i \in \mathcal{V}^s$ stands for a vector of which the i-th component is equal to 1 and all other components are equal to 0. In a number of cases, the indices are obviously cyclic. In those cases, indices k exceeding s should be read as (k mod s). For a set $\mathcal{S}$, the symbol $|\mathcal{S}|$ stands for the cardinality of $\mathcal{S}$, and for an event E the symbol $I\{E\}$ stands for the indicator function on E. The expectation of a random variable X, if well-defined, is denoted by $\mathrm{E}X$ or, if convenient, by $\mathrm{E}\{X\}$. Abbreviations that are used in several chapters are introduced once per chapter.

Chapter 2

The power-series algorithm

2.1 Introduction

The power-series algorithm (PSA) is an algorithm to compute the steady-state distribution of multiple-queue systems that can be modeled as a multi-dimensional quasi birth-and-death (QBD) process. Such a process consists of a vector of random variables describing the number of customers present at each queue, and possibly a supplementary vector (with finite range) to model for example non-exponentiality of the arrival and service process. The basic idea of the PSA is the transformation of the non-recursively solvable (infinite) set of balance equations into an, in principle, recursively solvable set of equations by adding one dimension to the state space. This transformation is realized by expressing the state probabilities as power series in some variable in light traffic. The first part of this chapter gives a survey of various aspects of the use of the PSA for the analysis of QBD processes (cf. also [22], [24]). In the second part we extend the PSA to the computation of derivatives, making the PSA more easily applicable for optimization purposes.

The ultimate goal of system modeling and analysis is efficient operation and system optimization. Optimization procedures involving real-valued decision variables generally require (partial) derivatives of a function, the cost function, to be optimized. When these partial derivatives are estimated on the basis of finite differences (cf. e.g. section 6.7 of [141], section 4.6 of [91]), one is confronted with a number of practical difficulties.
Firstly, it is a priori unclear which step size should be chosen to calculate these finite differences, while it is known that the choice of an appropriate step size, say h, may have a large impact on the efficiency of the optimization procedure. More precisely, when h is too large, higher order derivatives of the cost function may predominate in the estimation of the derivatives. On the other hand, if h is too small the numerator and the denominator of the gradient estimator may both be close to zero, making the estimated derivative highly sensitive to

inaccuracies in the computed values of the cost function. However, such inaccuracies are unavoidable if the cost-function values are not known exactly and numerical algorithms have to be applied to compute these values. Numerical experience with the PSA has indicated that an inappropriate choice of the step size may have a dramatical effect on the computation time of an optimization procedure, and in many cases the optimum is not found at all. These observations make optimization procedures in which derivatives are estimated by finite differences unreliable.
Secondly, the performance of neighboring schedules has to be evaluated, which may be a rather time-consuming task.
Thirdly, for parameter values nearby or at the boundary of the feasible region, neighboring values may be infeasible, so that modifications of the finite-difference estimator would have to be made. In practice, the latter goes at the expense of the transparency of the computer program of the optimization algorithm, because many 'special cases' have to be dealt with.

To be exempted from these practical complications, we extend the use of the PSA to the computation, instead of estimation, of derivatives. In practice, this extension makes the PSA much more easily applicable for optimization purposes. A wide variety of numerical optimization problems related to queueing models can be solved by combining the extension of the PSA with some classical gradient method for non-linear optimization (cf. e.g. chapter 6 of [141] for an overview). On the negative side, the extension of the PSA to the computation of derivatives increases the required amount of storage capacity. We refer to section 2.4 for a discussion about the computational complexity of the extension of the PSA and for practical notes on the implementation.
We present a general approach for the computation of derivatives of performance measures for a broad class of system parameters. For parameters which are controllable (e.g. routing probabilities for servers and/or customers) the extension allows the use of gradient methods to optimize the expected performance. For parameters which do not serve as decision variables (e.g. arrival rates) the extension is useful for the analysis of the sensitivity of performance measures with respect to these system parameters.

The remainder of this chapter is organized as follows. In section 2.2 a detailed description of the general QBD model is given. In section 2.3 various aspects of the PSA for this general model are extensively discussed. In section 2.4 the applicability of the PSA is extended to the computation of derivatives of the state probabilities with respect to a general class of continuous system parameters. Section 2.5 contains some concluding remarks.

2.2 Model description

Consider a multiple-queue model consisting of s queues, $Q_1, \ldots, Q_s$. The joint queue-length process is described by an s-dimensional vector $\boldsymbol{N}(t) =$

$(N_1(t),\ldots,N_s(t))$, which indicates the number of customers at each of the queues at time t, $t \geq 0$. In general, the process $\{\boldsymbol{N}(t),\ t \geq 0\}$ is not a Markov process. To transform the process $\{\boldsymbol{N}(t), t \geq 0\}$ into a Markov process, we add a vector of supplementary variables $\boldsymbol{\Phi}(t)$. This variable may, for instance, be used to model phase-type distributions in the arrival or service processes, or to model the status of the servers. For simplicity of the discussion, it is assumed that the supplementary space is the same for each $\boldsymbol{n} \in \mathbb{N}^s$, while it is possible that some states $(\boldsymbol{n},\boldsymbol{\varphi})$ can not be entered. The supplementary space is assumed to be finite and is denoted by $\mathcal{S}$. The joint process $\{(\boldsymbol{N}(t),\boldsymbol{\Phi}(t)), t \geq 0\}$ is a Markov process (on the state space $\mathbb{N}^s \times \mathcal{S}$) with a QBD structure; that is, the time between an entrance at state $(\boldsymbol{n},\boldsymbol{\varphi})$ and the successive departure from that state is exponentially distributed and transitions are only possible to states with at most one unit more or one unit less in one of the first s entries. The one-step transition rates are defined as follows: for $(\boldsymbol{n},\boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$, $\boldsymbol{\psi} \in \mathcal{S}$,

$\chi a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$: the arrival rate at Q_j at state $(\boldsymbol{n},\boldsymbol{\varphi})$, leading to a transition to state $(\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi})$, $j = 1,\ldots,s$;

$d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$: the departure rate from Q_j at state $(\boldsymbol{n},\boldsymbol{\varphi})$, leading to a transition to state $(\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi})$, with $d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})=0$ if $n_j = 0$, $j = 1,\ldots,s$;

$u(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$: the phase-transition rate from state $(\boldsymbol{n},\boldsymbol{\varphi})$ to $(\boldsymbol{n},\boldsymbol{\psi})$.

For $(\boldsymbol{n},\boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$, $\boldsymbol{\psi} \in \mathcal{S}, \chi \geq 0$, $j = 1,\ldots,s$, write

$$\lambda^{(j)}(\chi;\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}) := \chi a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}). \tag{2.1}$$

For a given set of relative arrival rates $a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$, $(\boldsymbol{n},\boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}, \boldsymbol{\psi} \in \mathcal{S}$, $j = 1,\ldots,s$, the arrival rates $\lambda^{(j)}(\chi;\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$, $\chi \geq 0$, $(\boldsymbol{n},\boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}, \boldsymbol{\psi} \in \mathcal{S}$, $j = 1,\ldots,s$, are functions of χ. In this way, the parametrization (2.1) characterizes a *family of systems*, each of which corresponds uniquely to one particular value of χ. The quantity χ will be used as a variable in the PSA.

Although necessary and sufficient conditions for the stability of the system are generally unknown, it is assumed that there exists a real-valued positive χ^* such that the system is stable for $0 \leq \chi < \chi^*$. In the sequel it is assumed that this condition is satisfied and that the system is in steady state. Denote by $(\boldsymbol{N},\boldsymbol{\Phi})$ stochastic variables with as joint distribution the stationary distribution of $(\boldsymbol{N}(t),\boldsymbol{\Phi}(t))$.
The transition rates $a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$, $d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$ and $u(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$ are assumed to be functions of some vector of continuous control variables $\boldsymbol{\gamma} = (\gamma_1,\ldots,\gamma_R)$. It is assumed that they are differentiable with respect to γ_r; the partial derivatives are denoted by $a_r^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$, $d_r^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$ and $u_r(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$, respectively, $r = 1,\ldots,R$. Note that some of these derivatives may vanish. The variable χ is assumed to be independent of $\boldsymbol{\gamma}$. The components of $\boldsymbol{\gamma}$ may be, for instance, arrival rates, service rates, routing probabilities (for servers or customers) or parameters of a service discipline.

2.3 The power-series algorithm for quasi birth-and-death processes

In this section we demonstrate how, under rather weak assumptions, the PSA can be applied to the general QBD model discussed in the previous section. The state probabilities are defined and the global balance equations are formulated. Then, the state probabilities are expressed as power series and a set of linear relations between the coefficients of these power series is obtained. Finally, a partial ordering is defined to compute the coefficients of the power series mainly recursively.

2.3.1 Balance equations

Define the state probabilities as follows: for $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, \boldsymbol{\varphi}) = \Pr\{(\boldsymbol{N}, \boldsymbol{\Phi}) = (\boldsymbol{n}, \boldsymbol{\varphi})\}. \tag{2.2}$$

The ergodicity assumption implies that for each state $(\boldsymbol{n}, \boldsymbol{\varphi})$ the total rate into that state is equal to the total rate out of that state. State transitions occur at instants of either a customer arrival, or a service completion of a customer, or a phase transition. The rate-in-rate-out equations for the state probabilities (2.2) can be formulated as follows: for $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$,

$$\begin{aligned}
&\left(\sum_{j=1}^{s} \sum_{\boldsymbol{\psi} \in \mathcal{S}} \left[\chi a^{(j)}(\boldsymbol{n}, \boldsymbol{\varphi}, \boldsymbol{\psi}) + d^{(j)}(\boldsymbol{n}, \boldsymbol{\varphi}, \boldsymbol{\psi}) \right] \right. \\
&\qquad \left. + \sum_{\boldsymbol{\psi} \in \mathcal{S}} u(\boldsymbol{n}, \boldsymbol{\varphi}, \boldsymbol{\psi}) \right) p(\boldsymbol{n}, \boldsymbol{\varphi}) = \\
&\chi \sum_{j=1}^{s} \sum_{\boldsymbol{\psi} \in \mathcal{S}} a^{(j)}(\boldsymbol{n} - \boldsymbol{e}_j, \boldsymbol{\psi}, \boldsymbol{\varphi}) p(\boldsymbol{n} - \boldsymbol{e}_j, \boldsymbol{\psi}) I\{n_j > 0\} \\
&+ \sum_{j=1}^{s} \sum_{\boldsymbol{\psi} \in \mathcal{S}} d^{(j)}(\boldsymbol{n} + \boldsymbol{e}_j, \boldsymbol{\psi}, \boldsymbol{\varphi}) p(\boldsymbol{n} + \boldsymbol{e}_j, \boldsymbol{\psi}) \\
&+ \sum_{\boldsymbol{\psi} \in \mathcal{S}} u(\boldsymbol{n}, \boldsymbol{\psi}, \boldsymbol{\varphi}) p(\boldsymbol{n}, \boldsymbol{\psi}).
\end{aligned} \tag{2.3}$$

Further, according to the law of total probability we have

$$\sum_{(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}} p(\boldsymbol{n}, \boldsymbol{\varphi}) = 1. \tag{2.4}$$

In general, (2.3) and (2.4) form an infinite set of equations that can not be solved recursively. The PSA transforms this set of equations into a mainly recursively solvable set of equations by expressing the state probabilities as power series in the variable χ at the origin. We will first discuss the conditions under which this technique is applicable.

2.3.2 Conditions for application of the algorithm

The solution method for the set of equations (2.3), (2.4) relies on the following property: for $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, \boldsymbol{\varphi}) = O(\chi^{|\boldsymbol{n}|}), \quad \chi \downarrow 0. \tag{2.5}$$

This property (2.5) can be shown to be valid under the following conditions (cf. [166]): for each state $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}, \boldsymbol{n} \neq \mathbf{0}$, either $p(\boldsymbol{n}, \boldsymbol{\varphi}) = 0$, or there exists a path $\boldsymbol{\varphi}^{(0)}, \boldsymbol{\varphi}^{(1)}, \ldots, \boldsymbol{\varphi}^{(\nu)}$ in $\mathcal{S}$ for some ν, $0 \leq \nu < |\mathcal{S}|$, such that

$$\boldsymbol{\varphi}^{(0)} = \boldsymbol{\varphi}, \; u(\boldsymbol{n}, \boldsymbol{\varphi}^{(i-1)}, \boldsymbol{\varphi}^{(i)}) > 0, \; i = 1, \ldots, \nu, \text{ and} \tag{2.6}$$

$$\sum_{j=1}^{s} \sum_{\boldsymbol{\psi} \in \mathcal{S}} d^{(j)}(\boldsymbol{n}, \boldsymbol{\varphi}^{(\nu)}, \boldsymbol{\psi}) > 0. \tag{2.7}$$

Thus, property (2.5) is valid if for each reachable $\boldsymbol{n}, \boldsymbol{n} \neq \mathbf{0}$, there is at least one positive departure rate, and for each reachable state $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}, \boldsymbol{n} \neq \mathbf{0}$, the probability that a departure occurs before any arrival takes place, after the process has entered this state, is positive. This assertion relies on induction to the sum of the queue lengths, $|\boldsymbol{n}|$, and to the length of the path, ν (cf. [168]). Conditions (2.6) and (2.7) are fulfilled in many practical cases, but it is *not*, for instance, if service only starts when the number of customers in a queue has reached some *threshold* larger than 1.
It should be noted that although conditions (2.6) and (2.7) can not hold for empty states $(\mathbf{0}, \boldsymbol{\varphi})$, $\boldsymbol{\varphi} \in \mathcal{S}$, property (2.5) trivially holds for $\boldsymbol{n}$=$\mathbf{0}$. However, these states require a special treatment, as will be shown in the next section.

2.3.3 Computational scheme

Based on property (2.5), we introduce the following formal power-series expansions for the state probabilities: for $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$, $0 \leq |\chi| < \chi_0$,

$$p(\boldsymbol{n}, \boldsymbol{\varphi}) = \chi^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \chi^k b_0(k; \boldsymbol{n}, \boldsymbol{\varphi}), \tag{2.8}$$

for some positive real-valued radius of convergence χ_0. We refer to Van den Hout and Blanc [167], [166] and Hooghiemstra et al. [97] for conditions under which there exists a positive χ_0 such that the power series converge for all $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$. Substituting the power-series expansions (2.8) into the global balance equations (2.3) implies the following set of equations for the coefficients $b_0(k; \boldsymbol{n}, \boldsymbol{\varphi})$: for $(k; \boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$, $0 \leq |\chi| < \chi_0$,

$$\chi^{|\boldsymbol{n}|}\sum_{k=0}^{\infty}\chi^k\left(\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}\left[\chi a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})+d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})\right]+\sum_{\boldsymbol{\psi}\in\mathcal{S}}u(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})\right)b_0(k;\boldsymbol{n},\boldsymbol{\varphi})=$$
$$\chi^{|\boldsymbol{n}|-1}\sum_{k=0}^{\infty}\chi^k\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}\chi a^{(j)}(\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi})I\{n_j>0\} \quad (2.9)$$
$$+\chi^{|\boldsymbol{n}|+1}\sum_{k=0}^{\infty}\chi^k\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}d^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi})$$
$$+\chi^{|\boldsymbol{n}|}\sum_{k=0}^{\infty}\chi^k\sum_{\boldsymbol{\psi}\in\mathcal{S}}u(\boldsymbol{n},\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n},\boldsymbol{\psi}).$$

Eliminating the factor $\chi^{|\boldsymbol{n}|}$ from both sides of this set of equations (2.9) yields: for $(\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^s\times\mathcal{S}$, $0\le|\chi|<\chi_0$,

$$\sum_{k=0}^{\infty}\chi^{k+1}\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$$
$$+\sum_{k=0}^{\infty}\chi^k\sum_{\boldsymbol{\psi}\in\mathcal{S}}\left(\sum_{j=1}^{s}d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})+u(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})\right)b_0(k;\boldsymbol{n},\boldsymbol{\varphi})=$$
$$\sum_{k=0}^{\infty}\chi^k\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}a^{(j)}(\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi})I\{n_j>0\} \quad (2.10)$$
$$+\sum_{k=0}^{\infty}\chi^{k+1}\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}d^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi})$$
$$+\sum_{k=0}^{\infty}\chi^k\sum_{\boldsymbol{\psi}\in\mathcal{S}}u(\boldsymbol{n},\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n},\boldsymbol{\psi}).$$

Both sides of the set of equations (2.10) represent functions of χ, $0\le|\chi|<\chi_0$. As a consequence, the coefficients of corresponding powers of χ are equal. Equating the coefficients of the k-th powers of χ yields: for $(k;\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^{1+s}\times\mathcal{S}$,

$$\sum_{\boldsymbol{\psi}\in\mathcal{S}}\left(\sum_{j=1}^{s}d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})+u(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})\right)b_0(k;\boldsymbol{n},\boldsymbol{\varphi})=$$
$$\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}a^{(j)}(\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi})I\{n_j>0\}$$
$$-\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})b_0(k-1;\boldsymbol{n},\boldsymbol{\varphi})I\{k>0\} \quad (2.11)$$
$$+\sum_{j=1}^{s}\sum_{\boldsymbol{\psi}\in\mathcal{S}}d^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi})I\{k>0\}$$
$$+\sum_{\boldsymbol{\psi}\in\mathcal{S}}u(\boldsymbol{n},\boldsymbol{\psi},\boldsymbol{\varphi})b_0(k;\boldsymbol{n},\boldsymbol{\psi}).$$

This set of equations (2.11) forms a recursive scheme with respect to the components $(k;\boldsymbol{n})$ under the following partial ordering $\prec$ of the vectors $(k;\boldsymbol{n},\boldsymbol{\varphi})$: for $(k;\boldsymbol{n},\boldsymbol{\varphi}),(\hat{k};\hat{\boldsymbol{n}},\hat{\boldsymbol{\varphi}})\in\mathbb{N}^{1+s}\times\mathcal{S}$,

$$(k;\boldsymbol{n},\varphi) \prec (\hat{k};\hat{\boldsymbol{n}},\hat{\varphi})$$
$$\text{if } \left[k+|\boldsymbol{n}| < \hat{k}+|\hat{\boldsymbol{n}}|\right] \vee \left[k+|\boldsymbol{n}| = \hat{k}+|\hat{\boldsymbol{n}}| \wedge k < \hat{k}\right]. \tag{2.12}$$

It can be readily verified that (2.11) expresses the coefficients $b_0(k;\boldsymbol{n},\varphi)$, with $(k;\boldsymbol{n},\varphi) \in \mathbb{N}^{1+s} \times \mathcal{S}$, in terms of coefficients of lower order than $(k;\boldsymbol{n},\varphi)$ with respect to $\prec$, except for the coefficients $b_0(k;\boldsymbol{n},\psi)$, $\psi \in \mathcal{S}$. Hence, the coefficients can be calculated (mainly) *recursively* in increasing order with respect to $\prec$, where for each $(k;\boldsymbol{n})$ a set of at most $|\mathcal{S}|$ linear equations may have to be solved.

The PSA is most efficient when the coefficients $b_0(k;\boldsymbol{n},\varphi)$, $\varphi \in \mathcal{S}$, can be computed *fully* recursively. Such a recursive scheme can be obtained if for each $\boldsymbol{n} \in \mathbb{N}^s$ the supplementary space $\mathcal{S}$ can be ordered in such a way that transitions without $\boldsymbol{N}$ leaving the state $\boldsymbol{n}$ are only possible in one direction. Therefore, Coxian distributions are usually easier to handle than more general phase-type distributions (cf. also section 2.5).

The same conditions (2.6), (2.7) which guarantee that (2.5) holds also guarantee that these sets of equations possess a unique solution (cf. [167] for a more detailed discussion). The only exceptions are formed by empty states, i.e. states with $\boldsymbol{n} = \boldsymbol{0}$. In these cases all departure rates vanish, so that the sets of equations (2.11) reduce to: for $\varphi \in \mathcal{S}$, $k = 0, 1, \ldots$,

$$\sum_{\psi\in\mathcal{S}} u(\boldsymbol{0},\varphi,\psi) b_0(k;\boldsymbol{0},\varphi) = \sum_{\psi\in\mathcal{S}} u(\boldsymbol{0},\psi,\varphi) b_0(k;\boldsymbol{0},\psi) + y_0(k;\varphi), \tag{2.13}$$

where

$$\begin{aligned} &y_0(k;\varphi) := \\ &-\sum_{j=1}^{s} \sum_{\psi\in\mathcal{S}} a^{(j)}(\boldsymbol{0},\varphi,\psi) b_0(k-1;\boldsymbol{0},\varphi) I\{k>0\} \\ &+\sum_{j=1}^{s} \sum_{\psi\in\mathcal{S}} d^{(j)}(\boldsymbol{e}_j,\psi,\varphi) b_0(k-1;\boldsymbol{e}_j,\psi) I\{k>0\}. \end{aligned} \tag{2.14}$$

One may verify by summing the equations (2.13) over φ, $\varphi \in \mathcal{S}$, that these are *dependent* sets of equations for the coefficients $b_0(k;\boldsymbol{0},\varphi)$, $\varphi \in \mathcal{S}$, for each k, $k = 0, 1, \ldots$. In addition, a necessary balance between the empty states and the states with one customer in the system implies that: for $k = 0, 1, \ldots$,

$$\begin{aligned} &\sum_{\varphi\in\mathcal{S}} y_0(k;\varphi) = \\ &-\sum_{j=1}^{s} \sum_{\varphi\in\mathcal{S}} \sum_{\psi\in\mathcal{S}} a^{(j)}(\boldsymbol{0},\varphi,\psi) b_0(k-1;\boldsymbol{0},\varphi) I\{k>0\} \\ &+\sum_{j=1}^{s} \sum_{\varphi\in\mathcal{S}} \sum_{\psi\in\mathcal{S}} d^{(j)}(\boldsymbol{e}_j,\psi,\varphi) b_0(k-1;\boldsymbol{e}_j,\psi) I\{k>0\} \\ &= 0, \end{aligned} \tag{2.15}$$

so that the dependent set of equations is not contradictory. An additional equation between the coefficients $b_0(k;\boldsymbol{n},\boldsymbol{\varphi})$, $(k;\boldsymbol{n},\boldsymbol{\varphi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$, follows from the law of total probability (2.4):

$$\sum_{\boldsymbol{\varphi}\in\mathcal{S}} b_0(0;\mathbf{0},\boldsymbol{\varphi}) = 1; \tag{2.16}$$

$$\sum_{\boldsymbol{\varphi}\in\mathcal{S}} b_0(k;\mathbf{0},\boldsymbol{\varphi}) = -\sum_{0<|\boldsymbol{n}|\le k}\sum_{\boldsymbol{\psi}\in\mathcal{S}} b_0(k-|\boldsymbol{n}|;\boldsymbol{n},\boldsymbol{\psi}), \quad k = 1,2,\ldots . \tag{2.17}$$

Note that the right-hand side of (2.17) consists of terms of lower order than $b_0(k;\mathbf{0},\boldsymbol{\varphi})$ with respect to $\prec$.
In the sequel, it is assumed that all but one of the equations (2.11) together with either (2.16) or (2.17) determine $b_0(k;\mathbf{0},\boldsymbol{\varphi})$, $\boldsymbol{\varphi} \in \mathcal{S}$, $k = 0,1,\ldots$. One may verify that the left-hand side coefficients of these sets of equations are the same for all k, so that the sets of equations are solvable for all k, $k = 0,1,\ldots$, if and only if they are solvable for $k = 0$. A *sufficient* condition for the solvability of the set of equations is that the Markov chain with transition rates $u(\mathbf{0},\boldsymbol{\varphi},\boldsymbol{\psi})$, $\boldsymbol{\varphi},\boldsymbol{\psi} \in \mathcal{S}$, is irreducible. In other words, the set of equations is uniquely solvable if the process, conditioned on the event that $\boldsymbol{N} = \mathbf{0}$ and no arrivals occur at all, is irreducible on the subset of $\mathcal{S}$ of reachable states (cf. [167] for a more detailed discussion). Throughout, this conditioned process will be referred to as the **0**-process. If the **0**-process has more than one recurrent class, the order in which the coefficients of the power-series expansions are computed has to be modified. Chapter 5 contains an extensive discussion of an example of how the PSA can be modified in such cases.

The PSA allows the computation of all state probabilities and hence, of any real-valued function of the state-probabilities. In practice, only a limited number of performance measures, say L, may have to be computed (e.g. mean queue lengths), rather than all individual state probabilities. Let $g^{(l)}(\boldsymbol{n},\boldsymbol{\varphi})$ be an arbitrary real-valued function of the state space. Most performance measures of the system are of the form $\mathrm{E}\left\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\right\}$, $l = 1,\ldots,L$. These general performance measures can be expressed in terms of the coefficients $b_0(k;\boldsymbol{n},\boldsymbol{\varphi})$ as follows: for $l = 1,\ldots,L$,

$$\mathrm{E}\left\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\right\} = \sum_{k=0}^{\infty} \chi^k f^{(l)}(k), \tag{2.18}$$

where for $k = 0,1,\ldots$,

$$f^{(l)}(k) := \sum_{0\le|\boldsymbol{n}|\le k}\sum_{\boldsymbol{\varphi}\in\mathcal{S}} g^{(l)}(\boldsymbol{n},\boldsymbol{\varphi}) b_0(k-|\boldsymbol{n}|;\boldsymbol{n},\boldsymbol{\varphi}). \tag{2.19}$$

In practice, only a finite number of coefficients can be computed (because of limitations on the available amounts of computation time and storage capacity). Let M be the number of terms that one wants or has to compute.

The following computational scheme shows how the performance measures $E\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\}$, $l=1,\ldots,L$, can be computed:

step 1 : let $f^{(l)}(k) := 0$, $l=1,\ldots,L$, $k=0,1,\ldots,M$;

step 2 : determine $b_0(0;\mathbf{0},\boldsymbol{\varphi})$, $\boldsymbol{\varphi}\in\mathcal{S}$, by solving all but one of the equations (2.13) together with (2.16), and update $f^{(l)}(0)$, $l=1\ldots,L$, according to (2.19);

step 3 : $m := 1$;

step 4 : for all $(k;\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^{1+s}\times\mathcal{S}$ with $\boldsymbol{n}\neq\mathbf{0}$ and with $k+|\boldsymbol{n}|=m$, determine $b_0(k;\boldsymbol{n},\boldsymbol{\varphi})$, according to (2.11) (in increasing order of $(k;\boldsymbol{n},\boldsymbol{\varphi})$ with respect to $\prec$), and update $f^{(l)}(m)$, $l=1,\ldots,L$, according to (2.19);

step 5 : determine $b_0(m;\mathbf{0},\boldsymbol{\varphi})$, $\boldsymbol{\varphi}\in\mathcal{S}$, by solving the set of equations consisting of all but one of the equations (2.11) together with (2.17), and update the value of $f^{(l)}(m)$, $l=1,\ldots,L$, according to (2.19);

step 6 : $m:=m+1$; if $m\leq M$ then return to *step 4*; otherwise STOP.

The global balance equations (2.3) depend on the parameters $a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$ and χ only through their product $\lambda^{(j)}(\chi;\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})=\chi a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})$, and consequently, so do the exact state probabilities (2.2). We will now show that this is also true when the state probabilities are only computed up to some finite power of χ. To this end, consider the following two different parametrizations of the arrival rates: for $(\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^s\times\mathcal{S}$, $\boldsymbol{\psi}\in\mathcal{S}$, $j=1,\ldots,s$,

$$\chi a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}) = \hat{\chi}\hat{a}^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}), \tag{2.20}$$

where

$$\hat{\chi}=c\chi, \quad \hat{a}^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}) = c^{-1}a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}), \tag{2.21}$$

for some scaling parameter $c>0$. Let the coefficients $\hat{b}_0(k;\boldsymbol{n},\boldsymbol{\varphi})$, $(k;\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^{1+s}\times\mathcal{S}$, be determined by equations (2.11), (2.16) and (2.17), with the convention that all rates $a^{(j)}(\cdot,\cdot,\cdot)$ are replaced by $\hat{a}^{(j)}(\cdot,\cdot,\cdot)$. Then one may verify by induction on the value of $k+|\boldsymbol{n}|$ that for all $(k;\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^{1+s}\times\mathcal{S}$,

$$\hat{b}_0(k;\boldsymbol{n},\boldsymbol{\varphi}) = \frac{b_0(k;\boldsymbol{n},\boldsymbol{\varphi})}{c^{k+|\boldsymbol{n}|}}. \tag{2.22}$$

As a consequence, when the state probabilities (2.8) are computed up to the M-th power of χ, with M finite, we have: for $(\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^s\times\mathcal{S}$, $M=0,1,\ldots$,

$$\begin{aligned}\hat{p}^{(M)}(\boldsymbol{n},\boldsymbol{\varphi}) &:= \hat{\chi}^{|\boldsymbol{n}|}\sum_{k=0}^{M}\hat{\chi}^k\hat{b}_0(k;\boldsymbol{n},\boldsymbol{\varphi})\\ &= (c\chi)^{|\boldsymbol{n}|}\sum_{k=0}^{M}(c\chi)^k\frac{b_0(k;\boldsymbol{n},\boldsymbol{\varphi})}{c^{k+|\boldsymbol{n}|}}\\ &= \chi^{|\boldsymbol{n}|}\sum_{k=0}^{M}\chi^k b_0(k;\boldsymbol{n},\boldsymbol{\varphi}),\end{aligned} \tag{2.23}$$

which corresponds to the partial sums of power-series expansions of the state probabilities in (2.8). This shows that the computed performance measure (up to a finite power χ) is indeed independent of the normalization of the arrival rates.

2.3.4 Convergence of the power series

The power series (2.8) are usually not convergent for all values of χ for which the system is stable. In the following discussion, we assume that the arrival rates are normalized such that the system is stable for $0 \leq \chi < 1$, and that the system becomes instable (in the case of infinite buffer sizes) when $\chi \uparrow 1$. In the literature, two different techniques are available to improve the convergence properties of the power series. The conformal mapping technique can be used to enlarge the radius of convergence of the power series by mapping singularities out of the unit circle. Alternatively, the epsilon algorithm can be applied to accelerate the convergence of slowly convergent series or to determine a value for divergent series by approximating the system performance measure under consideration by a sequence of quotients of polynomials. In this subsection we outline these two techniques.

Conformal mapping
The radius of convergence of the power series may be enlarged by introducing the following bilinear mapping of the interval [0,1] onto itself (cf. also [16], [22], [24]):

$$\theta = \Gamma_G(\chi) := \frac{(1+G)\chi}{1+G\chi}, \qquad \chi = \Gamma_G^{-1}(\theta) = \frac{\theta}{1+G-G\theta}, \quad G \geq 0. \tag{2.24}$$

Another computational scheme is then obtained by introducing power-series expansions of the state probabilities as functions of θ: for $(\boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^s \times \mathcal{S}$, $0 \leq |\theta| < \theta_0$,

$$p(\boldsymbol{n}, \boldsymbol{\varphi}) = \theta^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \theta^k \tilde{b}_0(k; \boldsymbol{n}, \boldsymbol{\varphi}), \tag{2.25}$$

where θ_0 is some strictly positive radius of convergence. This transformation maps possible singularities inside the region

$$|\chi - \frac{G}{1+2G}| > \frac{1+G}{1+2G}, \qquad |\chi| \leq 1, \tag{2.26}$$

outside the unit circle in the complex θ-plane. In this way any singularity outside the circle $|\chi - 1/2| = 1/2$ may be removed from the unit disk by an appropriate choice of the parameter G. Replacing χ by θ in the balance equations (2.3) according to (2.24), substituting the power series in θ into these equations and equating the coefficients of corresponding powers of θ in the resulting equations, leads to the following set of equations:

$$\sum_{\varphi\in\mathcal{S}} \tilde{b}_0(0;\mathbf{0},\varphi) = 1; \tag{2.27}$$

and

$$\sum_{\varphi\in\mathcal{S}} \tilde{b}_0(k;\mathbf{0},\varphi) = -\sum_{0<|\boldsymbol{n}|\le k}\sum_{\psi\in\mathcal{S}} \tilde{b}_0(k-|\boldsymbol{n}|;\boldsymbol{n},\psi), \quad k=1,2,\ldots\,; \tag{2.28}$$

and for $(k;\boldsymbol{n},\varphi)\in\mathbb{N}^{1+s}\times\mathcal{S}$, $\boldsymbol{n}\neq\mathbf{0}$,

$$\begin{aligned}
&(1+G)\sum_{\psi\in\mathcal{S}}\left(u(\boldsymbol{n},\varphi,\psi)+\sum_{j=1}^{s} d^{(j)}(\boldsymbol{n},\varphi,\psi)\right)\tilde{b}_0(k;\boldsymbol{n},\varphi) = \\
&\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} a^{(j)}(\boldsymbol{n}-\boldsymbol{e}_j,\psi,\varphi)\tilde{b}_0(k;\boldsymbol{n}-\boldsymbol{e}_j,\psi)I\{n_j>0\} \\
&-\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} a^{(j)}(\boldsymbol{n},\varphi,\psi)\tilde{b}_0(k-1;\boldsymbol{n},\varphi)I\{k>0\} \\
&+G\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} d^{(j)}(\boldsymbol{n},\varphi,\psi)\tilde{b}_0(k-1;\boldsymbol{n},\varphi)I\{k>0\} \\
&+G\sum_{\psi\in\mathcal{S}} u(\boldsymbol{n},\varphi,\psi)\tilde{b}_0(k-1;\boldsymbol{n},\varphi)I\{k>0\} \\
&+(1+G)\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} d^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\psi,\varphi)\tilde{b}_0(k-1;\boldsymbol{n}+\boldsymbol{e}_j,\psi)I\{k>0\} \\
&-G\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} d^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\psi,\varphi)\tilde{b}_0(k-2;\boldsymbol{n}+\boldsymbol{e}_j,\psi)I\{k>1\} \\
&+(1+G)\sum_{\psi\in\mathcal{S}} u(\boldsymbol{n},\psi,\varphi)\tilde{b}_0(k;\boldsymbol{n},\psi) \\
&-G\sum_{\psi\in\mathcal{S}} u(\boldsymbol{n},\psi,\varphi)\tilde{b}_0(k-1;\boldsymbol{n},\psi)I\{k>0\}\,.
\end{aligned} \tag{2.29}$$

Relation (2.29) differs from (2.11) mainly through the occurrence of terms with coefficients of the form $\tilde{b}_0(k-2;\boldsymbol{n}+\boldsymbol{e}_j,\psi)$. A priori, it is difficult to choose an appropriate value of the mapping parameter G, which depends on the generally unknown radii of convergence of the power series. The following rule of thumb for the choice of G is based on numerical experience (cf. also [24]). If only a few terms of the power series (say, 12–15) will or can be computed, take $G=0$; otherwise, execute a test run of $G=0$ and 5 to 10 terms, estimate the smallest radius of convergence among the computed power series of the performance measures $f^{(l)}(k)$, $l=1,\ldots,L$, and the computed state probabilities (cf. (2.19)). Then, take a value of G such that the power series are not too strongly divergent for the highest value of χ for which performance measures will be evaluated, to avoid numerical instabilities.

Epsilon algorithm

Another technique for removing singularities from inside the unit disk is the so-called epsilon algorithm. The epsilon algorithm aims to accelerate the convergence of slowly convergent sequences or to determine a value for divergent

sequences (cf. Wynn [174], Brezinsky [50]). The algorithm converts a polynomial into quotients of two polynomials, and is based on the following triangular recursive scheme: for $m = 0, 1, \ldots$, $\kappa = 0, 1, \ldots$,

$$\varepsilon_{\kappa+1}^{(m)} = \varepsilon_{\kappa-1}^{(m+1)} + \left[\varepsilon_{\kappa-1}^{(m+1)} - \varepsilon_{\kappa}^{(m)}\right]^{-1}, \quad \varepsilon_{-1}^{(m)} = 0, \quad \varepsilon_{0}^{(m)} = S_m; \tag{2.30}$$

here, the initial sequence S_m, $m = 0, 1, \ldots$, consists of partial sums of a series. The even sequences $\{\varepsilon_{2\kappa}^{(m)}, \ m = 0, 1, \ldots\}$, $\kappa = 1, 2, \ldots$, may converge faster to a limit than the initial sequence; the odd sequences are only intermediate steps in the calculation scheme. When S_m is the partial sum of a power series, say

$$S_m = \sum_{k=0}^{m} c_k \chi^k, \quad m = 0, 1, \ldots , \tag{2.31}$$

then the epsilon algorithm transforms this sequence of polynomials into sequences of quotients of two polynomials. More precisely, $\varepsilon_{2\kappa}^{(m-2\kappa)}$ will be a quotient of a polynomial of degree $m - \kappa$ over a polynomial of degree κ. Moreover, one may prove the following property (cf. [174]): for $\kappa = 1, 2, \ldots$, $m = 2\kappa, 2\kappa + 1, \ldots$,

$$|S_m - \varepsilon_{2\kappa}^{(m-2\kappa)}| = O(\chi^{m+1}), \quad \chi \to 0. \tag{2.32}$$

The epsilon algorithm turns a divergent series into a convergent series if the analytic continuation of the function defined by the series at $\chi = 0$ possesses only a finite number of poles as singularities inside the unit circle $|\chi| < 1$. The latter is, for instance, the case for Markov processes with a finite state space, in which case the state probabilities are rational functions of χ. In these cases the epsilon algorithm produces exact results after a finite number of steps (cf. [174]).

For queueing models for which the heavy-traffic behavior of the moments of the queue-length distribution is known beforehand, the performance of the epsilon algorithm can be strongly improved by a modification of the initial values $\varepsilon_0^{(m)}$. More specifically, in many queueing models it is known that moments of the queue-length distribution have a pole at $\chi = 1$ of a known order. For instance, if it is known that the performance measure under consideration has a first or second order pole at $\chi = 1$, then the initial values of the epsilon algorithm may be taken to be (cf. also [16], [18]):

$$\varepsilon_0^{(m)} = S_m + c_m \frac{\chi^{m+1}}{1 - \chi}, \quad m = 0, 1, \ldots , \tag{2.33}$$

and

$$\varepsilon_0^{(m)} = S_m + c_m \frac{\chi^{m+1}}{1 - \chi} + (c_m - c_{m-1}) \frac{\chi^{m+1}}{(1 - \chi)^2}, \quad m = 1, 2, \ldots , \tag{2.34}$$

instead of S_m, for a first or a second order pole, respectively. Numerical experience has taught us that application of the epsilon algorithm strongly improves the performance of the PSA and that in many cases it even leads to accurate estimation of heavy-traffic limits.

The number of terms M of the power-series expansions and the number of steps κ in the epsilon algorithm that are needed to reach a certain accuracy, depend on various properties of the model, such as the load of the system, the number of queues, the variations of distributions and the asymmetry between the system parameters. For most systems it is difficult to derive tight upper bounds on errors for the PSA together with the epsilon algorithm. The order of magnitude of the errors usually has to be estimated from differences in performance measures on the basis of M and of $M-1$, $M-2, \ldots$, terms of their power-series expansions. The reader is referred to Blanc [18], [22], Brezinsky [50], Koole [109], Wynn [174] and Baker and Graves-Morris [11], [12] for extensive discussions about the performance of the epsilon algorithm.
When the model under consideration is a polling model, so-called pseudo-conservation laws (cf. section 1.2.3) may give an indication of the accuracy of the computations.

2.3.5 Implementation

The main restriction in applying the PSA is the required amount of memory space. The total number of coefficients that has to be computed to determine the power series of the state probabilities up to the M-th power of χ is given by (cf. also [18], [22])

$$\begin{pmatrix} M+s+1 \\ s+1 \end{pmatrix} \times |\mathcal{S}|, \tag{2.35}$$

where the first factor stands for the number of couples $(k;\boldsymbol{n})$ for which $k+|\boldsymbol{n}| \leq M$. Thus, the memory requirements increase exponentially in the number of queues (s) and in the number of terms of the power series (M).

If coefficients are stored in *rectangular* arrays, then the required number of memory positions is equal to $(M+1)^{s+1}|\mathcal{S}|$. A strong reduction in the storage requirements can be achieved by mapping the $(s+1)$-dimensional region of lattice points $k+|\boldsymbol{n}| \leq M$ onto the set of integers by means of the one-to-one mapping (cf. [18])

$$C(k;\boldsymbol{n}) = \sum_{j=0}^{s} \begin{pmatrix} k+j+\sum_{i=1}^{j} n_i \\ j+1 \end{pmatrix}. \tag{2.36}$$

This *triangular* storage procedure enlarges the number of terms of the power-series expansions that can be computed with a given amount of storage capacity at the cost of increased computation time needed for the determination of the

location of the coefficients in the array in which they are stored.

A further reduction of the storage requirement can be achieved when only a limited number of performance measures (e.g. mean queue lengths) has to be evaluated, rather than all individual state probabilities. Then, the coefficients of the power-series expansions of the important performance measures can be *aggregated* during the execution of the PSA, and stored in small separate arrays (cf. (2.18), (2.19)), while the coefficients of the state probabilities can be removed as soon as they are not needed anymore in further computations. This *modulus* storage procedure strongly reduces the storage requirement in (2.35) for the calculation of the power-series expansions. Depending on the parameter G of the conformal mapping (as discussed in section 2.3.4) the storage requirements for the calculation of M terms of the power-series expansions are reduced to (cf. [18]): for $G = 0$,

$$\left(\begin{array}{c} M+s \\ s \end{array} \right) \times |\mathcal{S}|, \tag{2.37}$$

and for $G > 0$,

$$\left[\left(\begin{array}{c} M+s \\ s \end{array} \right) + \left(\begin{array}{c} M+s-2 \\ s-1 \end{array} \right) \right] \times |\mathcal{S}|. \tag{2.38}$$

The storage requirements for $G > 0$ exceed those for $G = 0$, because for $G > 0$ the coefficients $b_0(k-2; \boldsymbol{n} + \boldsymbol{e}_j, \psi)$ have to be kept in memory, as opposed to the case $G = 0$.
The ideas for efficient storage management given here generally lead to a considerable increase of the maximum number of terms M that can be computed for given amount of memory space, cf. Table 2 of [18] for an illustration. The increase in the efficiency of memory management has strongly enlarged the applicability of the PSA.

It is not easy to give general *rules of thumb* for the number of terms that has to be computed to achieve some desired degree of accuracy. This number generally depends on a number of factors such as the load offered to the system and on the 'degree of symmetry' of the model. If the model is 'fairly symmetrical' then 10 terms may suffice to give rather accurate results for lightly loaded systems; if the system is heavily loaded then 10 to 15 terms may still do well (by applying extrapolation techniques, cf. section 2.3.4). If the model is rather asymmetrical the algorithm may converge rather slowly. If the system is lightly loaded then 10 to 15 terms may still do well, but if the system is more heavily loaded then typically 30 or 40 terms (or even more) may be needed to achieve accurate results. We refer to section 2.4.2 for a discussion about the complexity of the PSA and for an indication of the maximal number of terms that can be computed for a given amount of storage capacity.

2.4 Extension to derivatives

In this section a complete computational scheme is derived to calculate the derivatives of performance measures with respect to control variables γ_r, $r = 1, \ldots, R$. This extension of the PSA is important, because it allows application of gradient methods to optimize the expected system performance and for analyzing the sensitivity of the system performance with respect to these control variables.

Define the derivatives of the state probabilities: for $r = 1, \ldots, R$, $(\boldsymbol{n}, \varphi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p_r(\boldsymbol{n}, \varphi) := \frac{\partial}{\partial \gamma_r} p(\boldsymbol{n}, \varphi). \tag{2.39}$$

These derivatives are expressed as power series in χ. For now, we assume that the variable χ satisfies the following two restrictions: (i) the system is stable for $0 \leq \chi < 1$, and (ii) the variable χ does *not* depend on the value of the control parameter $\boldsymbol{\gamma}$. At the end of this section we will show that the variable χ can indeed be chosen in such a way that these two properties are satisfied. Because χ is assumed to be independent of the control parameter $\boldsymbol{\gamma}$, the power-series expansions of the derivatives of the state probabilities with respect to the components of $\boldsymbol{\gamma}$ can be obtained by termwise differentiation of the power-series expansions of the state probabilities (2.8): for $r = 1, \ldots, R$, $(\boldsymbol{n}, \varphi) \in \mathbb{N}^s \times \mathcal{S}$, $0 \leq |\chi| < \chi_0$,

$$p_r(\boldsymbol{n}, \varphi) = \chi^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \chi^k b_r(k; \boldsymbol{n}, \varphi), \tag{2.40}$$

with

$$b_r(k; \boldsymbol{n}, \varphi) := \frac{\partial}{\partial \gamma_r} b_0(k; \boldsymbol{n}, \varphi). \tag{2.41}$$

2.4.1 Computational scheme

Differentiating both sides of the equations (2.11) with respect to $\gamma_r, r = 1, \ldots, R$, yields the following set of equations for the coefficients of the power series (cf. (2.8)): for $r = 1, \ldots, R$, $(k; \boldsymbol{n}, \varphi) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$
\begin{aligned}
&\sum_{\psi\in\mathcal{S}}\left(\sum_{j=1}^{s} d^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})+u(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})\right) b_r(k;\boldsymbol{n},\boldsymbol{\varphi}) = \\
&\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} a^{(j)}(\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi}) b_r(k;\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi}) I\{n_j>0\} \\
&+\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} a_r^{(j)}(\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi}) b_0(k;\boldsymbol{n}-\boldsymbol{e}_j,\boldsymbol{\psi}) I\{n_j>0\} \\
&-\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} a^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}) b_r(k-1;\boldsymbol{n},\boldsymbol{\varphi}) I\{k>0\} \\
&-\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} a_r^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi}) b_0(k-1;\boldsymbol{n},\boldsymbol{\varphi}) I\{k>0\} \\
&+\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} d^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi}) b_r(k-1;\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi}) I\{k>0\} \\
&+\sum_{j=1}^{s}\sum_{\psi\in\mathcal{S}} d_r^{(j)}(\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi}) b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_j,\boldsymbol{\psi}) I\{k>0\} \\
&+\sum_{\psi\in\mathcal{S}} \left(u(\boldsymbol{n},\boldsymbol{\psi},\boldsymbol{\varphi}) b_r(k;\boldsymbol{n},\boldsymbol{\psi})+u_r(\boldsymbol{n},\boldsymbol{\psi},\boldsymbol{\varphi}) b_0(k;\boldsymbol{n},\boldsymbol{\psi})\right) \\
&-\sum_{\psi\in\mathcal{S}}\left(\sum_{j=1}^{s} d_r^{(j)}(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})+u_r(\boldsymbol{n},\boldsymbol{\varphi},\boldsymbol{\psi})\right) b_0(k;\boldsymbol{n},\boldsymbol{\varphi}).
\end{aligned}
\tag{2.42}
$$

To derive a computational scheme for the coefficients $b_r(k;\boldsymbol{n},\boldsymbol{\varphi})$, we extend the partial ordering $\prec$ of the triples $(k;\boldsymbol{n},\boldsymbol{\varphi})$ in (2.12) to the partial ordering $\tilde{\prec}$ of the quadruples $(r,k;\boldsymbol{n},\boldsymbol{\varphi})$ in the following way: for $(r,k;\boldsymbol{n},\boldsymbol{\varphi}),(\hat{r},\hat{k};\hat{\boldsymbol{n}},\hat{\boldsymbol{\varphi}})\in\{0,1,\ldots,R\}\times\mathbb{N}^{1+s}\times\mathcal{S}$,

$$
\begin{aligned}
&(r,k;\boldsymbol{n},\boldsymbol{\varphi})\tilde{\prec}(\hat{r},\hat{k};\hat{\boldsymbol{n}},\hat{\boldsymbol{\varphi}}) \\
&\text{if}\quad [r=0\ \wedge\ \hat{r}>0]\ \vee\ \left[r=\hat{r}\ \wedge\ (k,\boldsymbol{n},\boldsymbol{\varphi})\prec(\hat{k},\hat{\boldsymbol{n}},\hat{\boldsymbol{\varphi}})\right].
\end{aligned}
\tag{2.43}
$$

The set of equations (2.42) expresses coefficients $b_r(k;\boldsymbol{n},\boldsymbol{\varphi})$, $r=0,1,\ldots,R$, $(k;\boldsymbol{n},\boldsymbol{\varphi})\in\mathbb{N}^{1+s}\times\mathcal{S}$, in terms of coefficients of lower order with respect to $\tilde{\prec}$, except for the terms $b_r(k;\boldsymbol{n},\boldsymbol{\psi})$, $\boldsymbol{\psi}\in\mathcal{S}$. Hence, the coefficients $b_r(k;\boldsymbol{n},\boldsymbol{\varphi})$ may be computed recursively in increasing order with respect to $\tilde{\prec}$, where for each $(r,k;\boldsymbol{n})$ a set of at most $|\mathcal{S}|$ linear equations, with unknowns $b_r(k;\boldsymbol{n},\boldsymbol{\varphi})$, $\boldsymbol{\varphi}\in\mathcal{S}$, may have to be solved. The only exceptions are formed by the states with $\boldsymbol{n}=\boldsymbol{0}$. In these cases the departure rates vanish, so that the equations (2.42) reduce to: for $r=1,\ldots,R$, $k=0,1,\ldots$, $\boldsymbol{\varphi}\in\mathcal{S}$,

$$
\sum_{\psi\in\mathcal{S}} u(\boldsymbol{0},\boldsymbol{\varphi},\boldsymbol{\psi}) b_r(k;\boldsymbol{0},\boldsymbol{\varphi}) = \sum_{\psi\in\mathcal{S}} u(\boldsymbol{0},\boldsymbol{\psi},\boldsymbol{\varphi}) b_r(k;\boldsymbol{0},\boldsymbol{\psi}) + y_r(k;\boldsymbol{\varphi}), \tag{2.44}
$$

where the quantities $y_r(k;\boldsymbol{\varphi})$, $r=1,\ldots,R$, $k=0,1,\ldots$, $\boldsymbol{\varphi}\in\mathcal{S}$, are defined by $y_r(0;\boldsymbol{\varphi}):=0$, and for $k=1,2,\ldots$, by

$$
\begin{aligned}
& y_r(k;\boldsymbol{\varphi}) := \\
& -\sum_{j=1}^{s} \sum_{\boldsymbol{\psi}\in\mathcal{S}} a^{(j)}(\mathbf{0},\boldsymbol{\varphi},\boldsymbol{\psi}) b_r(k-1;\mathbf{0},\boldsymbol{\varphi}) \\
& -\sum_{j=1}^{s} \sum_{\boldsymbol{\psi}\in\mathcal{S}} a_r^{(j)}(\mathbf{0},\boldsymbol{\varphi},\boldsymbol{\psi}) b_0(k-1;\mathbf{0},\boldsymbol{\varphi}) \\
& +\sum_{j=1}^{s} \sum_{\boldsymbol{\psi}\in\mathcal{S}} d^{(j)}(\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi}) b_r(k-1;\boldsymbol{e}_j,\boldsymbol{\psi}) \\
& +\sum_{j=1}^{s} \sum_{\boldsymbol{\psi}\in\mathcal{S}} d_r^{(j)}(\boldsymbol{e}_j,\boldsymbol{\psi},\boldsymbol{\varphi}) b_0(k-1;\boldsymbol{e}_j,\boldsymbol{\psi}) \\
& +\sum_{\boldsymbol{\psi}\in\mathcal{S}} \left(u_r(\mathbf{0},\boldsymbol{\psi},\boldsymbol{\varphi}) b_0(k;\mathbf{0},\boldsymbol{\psi}) - u_r(\mathbf{0},\boldsymbol{\varphi},\boldsymbol{\psi}) b_0(k;\mathbf{0},\boldsymbol{\varphi})\right).
\end{aligned}
\tag{2.45}
$$

All coefficients at the right-hand side of (2.45) are of lower order with respect to $\tilde{\prec}$ than $b_r(k;\mathbf{0},\boldsymbol{\varphi})$ and hence, can be considered to be known in (2.44). Because of a necessary balance in transitions between the set of empty states and the set of states with one customer in the system, one may verify by summing over $\boldsymbol{\varphi}$, $\boldsymbol{\varphi}\in\mathcal{S}$, that the set of equations (2.45) is a *dependent* set of equations for the coefficients $b_r(k;\mathbf{0},\boldsymbol{\varphi})$, $\boldsymbol{\varphi}\in\mathcal{S}$, for $r=1,\ldots,R$ and for $k=0,1,\ldots$. Termwise differentiation of relation (2.15) implies that this set of equations is not contradictory for $r=1,\ldots,R$.
An additional equation is implied by the law of total probability (2.4): for $r=1,\ldots,R$, it follows that for $k=0$,

$$
\sum_{\boldsymbol{\varphi}\in\mathcal{S}} b_r(0;\mathbf{0},\boldsymbol{\varphi}) = 0,
\tag{2.46}
$$

and that for $k=1,2,\ldots$,

$$
\sum_{\boldsymbol{\varphi}\in\mathcal{S}} b_r(k;\mathbf{0},\boldsymbol{\varphi}) = -\sum_{0<|\boldsymbol{n}|\le k} \sum_{\boldsymbol{\psi}\in\mathcal{S}} b_r(k-|\boldsymbol{n}|;\boldsymbol{n},\boldsymbol{\psi}).
\tag{2.47}
$$

The right-hand side of (2.47) contains only coefficients of states of a lower order with respect to $\tilde{\prec}$ than $b_r(k;\mathbf{0},\boldsymbol{\varphi})$. The left-hand side coefficients of the set of all but one of the equations (2.44) taken together with either (2.46) or (2.47) do not depend on k, $k=0,1,\ldots$, nor on r, $r=0,1,\ldots,R$ (for the case $r=0$ see (2.17)), so that it suffices to consider the solvability of the set of equations for $k=0$ and $r=0$. Hence, the set of equations (2.44) for the coefficients $b_r(k;\mathbf{0},\boldsymbol{\varphi})$, $\boldsymbol{\varphi}\in\mathcal{S}$, is uniquely solvable if and only if the set of equations (2.13) between the coefficients $b_0(0;\mathbf{0},\boldsymbol{\varphi})$, $\boldsymbol{\varphi}\in\mathcal{S}$, is uniquely solvable. The solvability of the latter set of equations has been discussed in section 2.3.3.

Let $\mathrm{E}\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\}$ be an arbitrary system performance measure that is a function of the state probabilities (cf. (2.18), (2.19)), and that is differentiable with respect to all components of the control variable $\boldsymbol{\gamma}$, $l=1,\ldots,L$. Then the derivative of $\mathrm{E}\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\}$ with respect to γ_r can be expressed as power series in χ by: for $r=1,\ldots,R$, $l=1,\ldots,L$,

$$\frac{\partial}{\partial \gamma_r} \mathrm{E}\left\{ g^{(l)}(\boldsymbol{N}, \boldsymbol{\Phi}) \right\} = \sum_{k=0}^{\infty} \chi^k f_r^{(l)}(k), \tag{2.48}$$

where

$$f_r^{(l)}(k) = \sum_{0 \leq |\boldsymbol{n}| \leq k} \sum_{\boldsymbol{\varphi} \in \mathcal{S}} g^{(l)}(\boldsymbol{n}, \boldsymbol{\varphi}) b_r(k - |\boldsymbol{n}|; \boldsymbol{n}, \boldsymbol{\varphi}). \tag{2.49}$$

For ease of the discussion it is assumed here that $g^{(l)}(\boldsymbol{n}, \boldsymbol{\varphi})$ does not depend on $\boldsymbol{\gamma}$, for $l = 1, \ldots, L$. Then the coefficients $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$, $r = 1, \ldots, R$, $(k; \boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$, of the power series in (2.8) and (2.40) can be computed according to the following computational scheme (up to the power M of χ):

step 1 : for $r = 1, \ldots, R$, let $f_r^{(l)}(k) := 0$, $l = 1, \ldots, L$, $k = 0, 1, \ldots, M$;

step 2 : for $r = 1, \ldots, R$, determine $b_r(0; \mathbf{0}, \boldsymbol{\varphi})$, $\boldsymbol{\varphi} \in \mathcal{S}$, by solving all but one of the equations (2.44) together with (2.46), and update $f_r^{(l)}(0), l = 1, \ldots, L$, according to (2.49);

step 3 : m:=1;

step 4 : for all $r = 1, \ldots, R$, $(k; \boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ with $\boldsymbol{n} \neq \mathbf{0}$ and with $k + |\boldsymbol{n}| = m$, determine $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$, according to (2.11) (in increasing order of $(r, k; \boldsymbol{n}, \boldsymbol{\varphi})$ with respect to $\tilde{\prec}$), and update the value of $f_r^{(l)}(m)$, $l = 1, \ldots, L$;

step 5 : for $r = 1, \ldots, R$, determine $b_r(m; \mathbf{0}, \boldsymbol{\varphi})$, $\boldsymbol{\varphi} \in \mathcal{S}$, by solving the set of equations consisting of all but one of the equations (2.44) together with (2.47), and update the value of $f_r^{(l)}(m)$, $l = 1, \ldots, L$;

step 6 : $m := m + 1$; if $m \leq M$ then return to *step 4* ; otherwise STOP.

Let us reconsider the normalization of the arrival rates. At the end of section 2.3.3 we showed that the choice of the normalization constraint on the arrival rates does not affect the computed system performance measures. Similarly, one may verify that by rescaling the variable χ with a factor c the coefficients $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$ are multiplied by a factor $c^{-(k+|\boldsymbol{n}|)}$ (cf. (2.22), (2.23)). With a similar derivation as in (2.23) it follows that the computed derivatives of the system performance measures (cf. (2.40)) remain unaltered.

Let us consider models in which the buffer sizes are infinite, so that a number of the queue lengths will explode when the arrival rates become very large. In section 2.2 we expressed the state probabilities as power series in χ, and in section 2.3.4 we assumed the variable χ to be normalized such that the system becomes instable for $\chi \uparrow 1$. Under this normalization, the variable χ generally depends on the value of $\boldsymbol{\gamma}$. We will now show that there exists a variable χ of

the power series which does *not* depend on γ and for which the system becomes instable as $\chi \uparrow 1$. To this end, let $\hat{\chi}$ be a power-series variable which does not depend on the value of the control parameter γ. Then the smallest value of $\hat{\chi}$ for which the system explodes depends on γ, say at $\hat{\chi} = f(\gamma)$, for some strictly positive real-valued function f. Because we consider the performance of the system for given values of γ, we fix the value of γ, say $\gamma = \gamma^{(0)}$. Then *for given* $\gamma = \gamma^{(0)}$, we define

$$\chi := \frac{\hat{\chi}}{f(\gamma^{(0)})}. \tag{2.50}$$

Then it is easily seen that the system becomes instable for $\chi \uparrow 1$. Moreover, from definition (2.50), χ is *not* a function of the variable γ, and differs from $\hat{\chi}$ only by a *constant* scale factor $f(\gamma^{(0)})$, which is known to have no influence on the computed performance measures and their derivatives. Note that the variable χ is normalized in such a way that the system explodes for $\chi \uparrow 1$ *only for* $\gamma = \gamma^{(0)}$. For neighboring values of γ, $\gamma \neq \gamma^{(0)}$, the system does not necessarily become instable for $\chi \uparrow 1$. These considerations motivate why χ, defined in (2.50), can be considered as a variable that is independent of the control parameter γ. In this way, one avoids complications which would occur if χ were chosen to be a (possibly non-differentiable) function of the variable γ.

2.4.2 Complexity

In section 2.3.5 we discussed the complexity of the PSA for the evaluation of performance measures. In this section we discuss the complexity of the PSA for the extension to the computation of derivatives.

The total number of coefficients that has to be computed to determine the power series of the state probabilities and their derivatives with respect to γ_r, $r = 1, \ldots, R$, up to the M-th power of χ is given by the following expression:

$$(R+1) \times \begin{pmatrix} M+s+1 \\ s+1 \end{pmatrix} \times |\mathcal{S}|. \tag{2.51}$$

Thus, the memory (and time) requirements increase *linearly* in the number of derivatives (R) that has to be computed.

In practice, one may be interested in only a limited number of performance measures. Hence, coefficients can be removed as soon as they are not needed anymore in further computations (as discussed in section 2.3.5), restricting the required amount of memory space given in (2.51) (for the case with conformal mapping parameter $G = 0$, cf. (2.37)) to

$$(R+1) \times \begin{pmatrix} M+s \\ s \end{pmatrix} \times |\mathcal{S}| \tag{2.52}$$

		$s=2$				$s=4$				$s=6$			
$\|\mathcal{S}\| \rightarrow$		4	12	24	48	4	12	24	48	4	12	24	48
R	0	2234	1289	911	643	85	64	53	44	31	25	22	19
$\downarrow$	1	1579	911	643	454	71	53	44	37	27	22	19	17
	2	1289	743	525	371	64	48	40	33	25	20	18	15
	5	911	525	371	262	53	40	33	27	22	18	15	13
	10	672	387	273	193	45	34	28	23	19	16	13	12

Table 2.1: Maximal number of terms at a storage capacity of 10^7 coefficients.

coefficients. As an illustration, we have computed the maximal number of terms of the power series that can be computed (according to (2.52)) for given amount of storage capacity of 10^7 coefficients and for various values of the number of queues (s), the size of the supplementary space ($|\mathcal{S}|$) and the number of derivatives (R). Table 2.1 shows that the number of terms of the power series that can be computed for a given amount of storage capacity may decrease considerably when the number of derivatives is increased. It should be noted that the case $R = 0$ corresponds to the situation in which no derivatives are computed (as discussed in section 2.3.3). When the conformal mapping parameter $G > 0$, the storage requirements are slightly increased (to $(R+1)$ times the expression in (2.38)). However, one may verify that the decrement of the number of terms that can be computed for given amount of memory space is at most 1. We refer to the discussion at the end of section 2.3.5 for rough guidelines for the number of terms of the power series that are needed to achieve some 'acceptable' degree of accuracy.

2.4.3 Implementation

We will now discuss some ideas about the implementation of the PSA which are specific for the extension to the computation of derivatives as elaborated upon in sections 2.4.1 and 2.4.2.

Firstly, it follows from the equations (2.42) that for given $(k; \boldsymbol{n}, \boldsymbol{\varphi})$ the coefficients $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$, $r = 1, \ldots, R$, can not be computed when $b_0(k; \boldsymbol{n}, \boldsymbol{\varphi})$ is unknown, but that the partial derivatives can be computed *separately*. It is most appropriate to compute for each $(k; \boldsymbol{n}, \boldsymbol{\varphi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ the coefficients $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$, $r \in \{1, \ldots, R\}$, *directly after* the computation of the coefficients $b_0(k; \boldsymbol{n}, \boldsymbol{\varphi})$. In this way the coefficients $b_0(k; \boldsymbol{n}, \boldsymbol{\varphi})$ need not be kept in memory.

Secondly, the fact that the coefficients $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$, $r = 1, \ldots, R$, can be computed separately also implies that one may *partition* the set $\mathcal{T} := \{1, \ldots, R\}$ into proper subsets, say $\mathcal{T}_1, \ldots, \mathcal{T}_m$ $(1 < m \leq R)$, and compute for each $(k; \boldsymbol{n}, \boldsymbol{\varphi})$ the coefficients $b_r(k; \boldsymbol{n}, \boldsymbol{\varphi})$ for $r \in \{0\} \cup \mathcal{T}_j$ *successively*, $j = 1, \ldots, m$. Note that

this partitioning may (partly) *compensate* for the loss of the maximal number of terms of the power series that can be computed (cf. Table 2.1). As an illustration of the partitioning, consider a model with $s = 6$, $|\mathcal{S}| = 12$ and $R = 5$ (so that derivatives with respect to 5 different system parameters have to be computed) and suppose the available amount of memory space is 10^7 coefficients. Then, according to Table 2.1, the maximal number of coefficients of the power series that can be computed is equal to 18. Suppose, however, that numerical experience with the PSA has taught us that in order to achieve an 'acceptable degree of accuracy' one needs to compute 20 terms of the power series. Then it follows from Table 2.1 that in order to have enough memory space to compute 20 terms of the power series the set $\mathcal{T} := \{1, \ldots, 5\}$ should be partitioned into sets of at most 2 elements. To this end, $\mathcal{T}$ could e.g. be partitioned into $\mathcal{T}_1 = \{1, 2\}$, $\mathcal{T}_2 = \{3, 4\}$, and $\mathcal{T}_3 = \{5\}$.
It should be noted that this partitioning leads to an increase of the required amount of computation time, because in this case the terms $b_0(k; \boldsymbol{n}, \boldsymbol{\varphi})$ have to be computed m times instead of once. The latter illustrates that in order to determine the gradient for given amount of memory space one has to deal with a *trade-off* between the required amount of computation time on the one hand and the accuracy of the computed performance measures on the other hand.

2.5 Concluding remarks

The PSA is applicable to models with a QBD structure. Therefore, general arrival processes such as the Markovian Arrival Processes (MAPs) can be handled. Service times and switch-over times may be of phase type, as introduced by Neuts [136]. Phase-type distributions can, in some sense, approximate any distribution function with non-negative support arbitrarily close (cf. [147]).

The PSA is most efficient when the coefficients can be determined recursively for each $(k; \boldsymbol{n})$-combination. For this reason, in the modeling of probability distributions, e.g. for service times, interarrival times or switch-over times, Coxian distributions are generally preferred to general phase-type distributions. The class of Coxian distributions lies dense in the class of all distribution with non-negative support (cf. [9]). However, in some cases the number of phases in a general phase-type distribution needed to approximate some probability distribution is considerably smaller than in the case of Coxian distributions. In those cases, phase-type distributions may be preferred to Coxian distributions.

When the extension of the PSA to the computation of derivatives is applied for optimization purposes, it is useful to apply the PSA with a small number of terms to find the neighborhood of the optimum with reduced computational effort, and then proceed with the PSA with more terms to locally improve the performance. We refer to section 3.6 for a more detailed discussion.

Recent developments have indicated that the PSA is also applicable to sys-

tems for which the QBD structure is violated. In Van den Hout and Blanc [167] the use of the PSA is extended to systems with Batch Markovian Arrival Processes (BMAPs). In [168] the PSA is further extended to the general class of so-called Markovian queueing networks, in which the arrival process is a Multi-queue Markovian Arrival Process (MMAP). Recently, Koole [109] has shown that the PSA is, in principle, applicable to general Markov processes. The computation of derivatives discussed in this chapter is readily applicable in the context of these generalizations of the PSA.

Higher-order derivatives can also be determined by means of the PSA, following similar lines as discussed above. Clearly, this further extension adds to the complexity of the PSA, limiting even more the number of terms that can be computed in power-series expansions.

Chapter 3

Optimization of polling systems with Bernoulli schedules

3.1 Introduction

In a polling system the server visits the queues according to some routing mechanism. In many situations, the server has no global information on the numbers of customers present at each of the queues. Therefore, often a static polling strategy is chosen for such systems. The major mechanisms for controlling static polling systems are service limits and time limits. Service limits restrict the number of customers served during one visit of the server to a queue. Time limits bound the maximal amount of time spent by the server at a queue. The standard control mechanisms, IEEE 802.5 Token Ring, IEEE 802.4 Token Bus and ANSI X3T9.5 FDDI all make use of service or time limits. The option of setting different limits to different queues can be used to give relative priority to some queues, while keeping some degree of fairness between the customers at the different queues. Although service limits and time limits are widely used control mechanisms, little is known about how to operate these mechanisms to achieve desired performance. In this chapter we investigate the proper choice of service limits. These limits are here assumed to be static, i.e. they have to be chosen a priori and can not be changed depending on the actual state of the system.
We will consider optimization of service limits within the class of Bernoulli service strategies (cf. section 1.3). The Bernoulli parameters serve as control variables. We consider the problem of finding a combination of Bernoulli parameters that minimizes a weighted sum of the mean waiting times at the various queues, where the weights corresponding to the different queues are set differently according to the relative importance of the queues.

The Bernoulli service discipline was introduced for the GI/G/1-Bernoulli vacation model by Keilson and Servi [102], and applied to polling models by Tedijanto [163]. The Bernoulli service discipline at queue i is characterized by a parameter q_i $(0 \leq q_i \leq 1)$. The class of Bernoulli service disciplines encompasses the classical 1-limited ($q_i = 0$) and exhaustive ($q_i = 1$) service disciplines as special cases. The Bernoulli service discipline does not satisfy the Additivity Property discussed in section 1.2.3 (except for the case $q_i = 1$). Detailed exact analysis is scarce and mainly restricted to systems with one or two queues.
For the GI/G/1-Bernoulli vacation model Keilson and Servi [102] show that the steady-state waiting time is the sum of two components, being (i) the waiting time in the GI/G/1 model with identical system parameters but without vacations, and (ii) the forward recurrence time of the vacation length. For models with Poisson arrivals there are some more exact results. For the M/G/1-Bernoulli vacation model, Ramaswamy and Servi [140] derive a closed-form expression for the waiting-time distribution. Two-queue models with exhaustive service at both queues ($q_1 = q_2 = 1$) are easy to analyze (cf. [157], [77]). Models with 1-limited service at one queue and exhaustive service at the other queue ($q_1 = 0$, $q_2 = 1$) are also rather easy to analyze (cf. [94]). Weststrate and Van der Mei [171] generalize these results by deriving the Laplace-Stieltjes Transforms (LSTs) of the waiting-time distributions in a two-queue model with Bernoulli service at one queue and exhaustive service at the other queue ($0 \leq q_1 \leq 1$, $q_2 = 1$) via an iterative procedure. Boxma and Groenendijk [41] use the technique of Riemann boundary-value problems to determine the waiting-time distributions in a two-queue model with 1-limited service at both queues ($q_1 = q_2 = 0$). Recently, for a two-queue model with Bernoulli service at both queues and zero switch-over times ($0 \leq q_1, q_2 \leq 1$), Lee [112] has formulated the problem of finding the waiting-time distributions as a Riemann boundary-value problem with a shift, and has solved the latter by exploring a Fredholm integral equation over the unit circle.

For polling systems with Bernoulli schedules with more than two queues, the most general result is the formulation of a pseudo-conservation law (PCL), i.e. an exact expression for a specific weighted sum of the mean waiting times (cf. [38], [40]). However, a more detailed analysis (e.g. on the individual mean waiting times) has turned out to be very intricate. Therefore, several methods to approximate expected system performance measures have been developed. Tedijanto [163] has proposed a mean waiting-time approximation based on the PCL for a cyclic polling model with a Bernoulli schedule and with an arbitrary number of queues. Based on the analysis of an M/G/1-Bernoulli vacation model, Servi [149] has proposed an iterative algorithm to approximate the mean waiting times at each of the queues. However, in many cases these approximation methods give rather inaccurate results.

Thus, a detailed analysis of polling models with Bernoulli schedules with more than one queue is generally hard to give. Moreover, if a detailed analysis is

possible at all, it may still be far from trivial to translate the results into numerical values for the system performance measures (such as in a number of two-queue models [41], [112]). As a consequence of this, there is a need for numerical algorithms for the analysis and optimization of these models. When service times and switch-over times are approximated by phase-type or Coxian distributions, the models considered here have a quasi birth-and-death (QBD) structure, so that they can be handled with the power-series algorithm (PSA), as elaborated upon extensively in chapter 2. In that chapter we have extended the use of the PSA to the computation of derivatives. In combination with some classical non-linear optimization procedure this extension of the PSA for the present model (with the computation of derivatives with respect to the Bernoulli parameters) provides a means to determine optimal combinations of Bernoulli parameters. This has enabled us to investigate properties of optimal Bernoulli schedules, and to analyze their sensitivity with respect to the parameters of the system.

This chapter concerns optimization of cyclic polling models with respect to the Bernoulli service disciplines at the queues. It is our aim to find a combination of Bernoulli parameters $(q_1^*, \ldots, q_s^*)$ that minimizes some arbitrary weighted sum of the mean waiting times at the queues. The optimization problem is partially analytically solvable. We derive some properties of optimal Bernoulli schedules.
Light-traffic limits of the mean waiting times are obtained by algebraically determining the first few terms of the power-series expansions of the mean waiting times. This leads to light-traffic limits for optimal Bernoulli schedules. For systems with non-zero switch-over times, heavy-traffic limits of optimal Bernoulli schedules follow directly from the stability condition, which forces the Bernoulli parameters to tend to one (i.e. exhaustive service) for each queue when the system load tends to one.
Finally, we obtain partially conjectured $c\mu$-like rules for optimal Bernoulli schedules. The rules state that all queues i for which the ratio c_i/ρ_i (cf. section 1.3) is maximal over all queues, should be served exhaustively (i.e. $q_i^* = 1$). Moreover, for models with zero switch-over times, the queues i for which this ratio is minimal over all queues should be served according to the 1-limited service discipline (i.e. $q_i^* = 0$). These rules restrict the dimension of the optimization problem and may solve the problem completely in some special cases. The components of the optimal schedule that are not covered by these rules can be determined numerically with the aid of the PSA. However, the computation time needed to find these components with the PSA may be significant, especially when the number of queues is large. Therefore, we propose a simple and fast-to-evaluate method to approximate optimal Bernoulli schedules. The proposed approximation method is tested extensively and is shown to be effective and accurate in optimizing the system performance.

The Bernoulli service discipline may be viewed as the stochastic counterpart of the classical limited service discipline, under which the number of customers

served during a visit of the server has a fixed, rather than a stochastic, upper bound. In practice, limited service disciplines are widely used service strategies. However, the analysis of polling models with limited service disciplines is even more involved than the analysis of models with Bernoulli service (cf. [172], [112]). This is due to the Markovian character of the Bernoulli service discipline (the service limits are geometrically distributed), as opposed to the limited service disciplines.

Blanc [19] uses the PSA to make a comparison between the performance of cyclic polling models with Bernoulli schedules $(q_1, \ldots, q_s)$ and models with limited service, with limits $(K_1, \ldots, K_s)$. His numerical experiments indicate that the mean waiting times in both systems pass globally through similar trajectories when the systems are considered as functions of one parameter (i.e. $K_j = K,\ q_j = q,\ j = 1, \ldots, s$).

Tedijanto [164] makes a comparison between the system performance of the Bernoulli vacation model (with parameter q) and the K-limited vacation model. He shows that the waiting time in the Bernoulli vacation model is stochastically larger (in the increased convex ordering sense) than in the vacation model with limited service (with $q = 1 - 1/K$), even in the transient regime.

In a recent study, Borst et al. [35] consider optimization of cyclic polling models with respect to the service disciplines at the queues. They consider cyclic polling models with limited service at each queue, where the (fixed) service limits serve as decision variables. An additional complicating factor here is that the service limits are integer valued, so that no standard non-linear continuous optimization procedures (e.g. in combination with the PSA) can be used. This makes the problem of finding optimal combinations of service limits even more involved compared to the optimization of Bernoulli schedules. In that perspective, the analysis of optimal combinations of Bernoulli parameters may be viewed as a starting point for optimization of limited service disciplines. In their paper, Borst et al. [35] propose simple approximations for optimal combinations of service limits for unconstrained optimization and for the constrained optimization problem in which a limit is put on the maximal total number of customers served during a cycle. They also use the PSA to search for optimal combinations of service limits, and obtain rules for optimal combinations of the service limits. Interestingly, they find that, although the performance of systems with Bernoulli service disciplines may differ rather strongly from that of systems with limited service disciplines (with $q_j = 1 - 1/K_j$, cf. [19]), the optimal Bernoulli schedule turns out to be a good emulation of the optimal combination of service limits under the limited service discipline.

The Bernoulli and the limited service discipline put some limit on the number of customers served during a visit of the server to a queue. Alternatively, the so-called time-limited service disciplines restrict the maximal duration of each visit of the server to a queue. Time-limited service disciplines are widely used in practice. Leung [115] analyzes polling models with time-limited service strategies with a numerical approach based on Discrete Fourier Transforms (cf. section 1.2.4). His numerical experiments suggest that the performance of models with time-limited service with exponentially distributed time limits

closely approximates systems with fixed time limits when the time limits are rather small. Moreover, he observes that for rather high time limits systems with fixed time-limited service behave rather similarly to customer-limited service. Polling models with time-limited service strategies can be analyzed and optimized with the aid of the PSA if the time limits have phase-type distributions.

The remainder of this chapter is organized as follows. Section 3.2 contains a brief description of the model. In section 3.3 we discuss how the model can be analyzed by means of the PSA, with the extension to the computation of derivatives with respect to the Bernoulli parameters. In section 3.4 we discuss properties of optimal Bernoulli schedules. Light- and heavy-traffic limits of optimal Bernoulli schedules are derived and a $c\mu$-like partial solution to the optimization problem is given. Section 3.5 contains an extensive discussion about the influence of the system parameters on optimal Bernoulli schedules. In section 3.6 we propose and test a simple and fast-to-evaluate method to approximate optimal Bernoulli schedules. Section 3.7 contains some concluding remarks.

3.2 Model description

We consider the basic polling model discussed in section 1.3 with s infinite-buffer queues, $Q_1, \ldots, Q_s$, with Poisson arrival streams with rate $\lambda_i = a_i\chi$ at Q_i. The service times at Q_i are Coxian distributed with parameters Ψ_i^1, $\pi_i^{1,\psi}$, $\mu_i^{1,\psi}$, $\psi = 1, \ldots, \Psi_i^1$. The switch-over times from Q_{i-1} to Q_i are Coxian distributed with parameters Ψ_i^0, $\pi_i^{0,\psi}$, $\mu_i^{0,\psi}$, $\psi = 1, \ldots, \Psi_i^0$, $i = 1, \ldots, s$. Denote by $\boldsymbol{\sigma}^{(k)} = (\sigma_1^{(k)}, \ldots, \sigma_s^{(k)})$ the vector of k-th moments of the switch-over times from Q_{i-1} to Q_i, and denote by σ_k the k-th moment of the total switch-over time per cycle of the server along the queues, k=1,2. Let $\eta_i := \rho_i/\rho$ denote the relative load offered to Q_i, $i = 1, \ldots, s$. The queues are served according to a Bernoulli schedule $\boldsymbol{q} = (q_1, \ldots, q_s)$. It should be noted that when the server finds a queue empty upon arrival, the server immediately proceeds to the next queue. Hence, if the system is entirely empty during some time interval, the server keeps on spinning around in the system.
The following conditions are necessary and sufficient for the stability of the system (cf. [85]):

$$\rho[1 + \sigma_1 a_i(1 - q_i)] < 1, \quad i = 1, \ldots, s. \tag{3.1}$$

In the sequel it is assumed that the stability condition (3.1) is satisfied and that the system is in steady state.

The optimization problem is to minimize the cost function, defined by

$$C(\boldsymbol{q}) := \sum_{i=1}^{s} c_i \mathrm{E}W_i, \tag{3.2}$$

over all $\boldsymbol{q}$ for which (3.1) holds. The components of $\boldsymbol{c} = (c_1, \ldots, c_s)$ are assumed to be strictly positive. Without loss of generality, it is assumed that the total sum of the components of $\boldsymbol{c}$ is equal to 1. The quantities EW_i, $i = 1, \ldots, s$, are the steady-state mean waiting times at the various queues, which depend on $\boldsymbol{q}$. An optimal Bernoulli schedule will be denoted by $\boldsymbol{q}^*$.

3.3 The power-series algorithm

Polling models with Bernoulli schedules are generally very hard to treat with mathematical techniques, except for a number of special cases, and so is the optimization of these systems with respect to the service limits. In this section we will show how the model discussed in the previous section can be analyzed by means of the PSA, with the extension to the computation of derivatives with respect to the Bernoulli parameters.
The model considered here has a QBD structure and also satisfies the conditions for application of the PSA. In principle, the approach discussed here is no more than a special case of the general description of the PSA given in the previous chapter. However, the specific structure of the present model is explored to compute the coefficients of the power-series expansions *completely recursively*. This makes the PSA much more efficient than in the general case, where for each $(k; \boldsymbol{n})$-combination a set of linear equations has to be solved (as many as the number of elements in the supplementary space, cf. section 2.3.3). Apart from this, it is interesting in its own right to see how the PSA can be applied to this specific model.

First, the state probabilities are defined and the global balance equations are formulated. Then, the state probabilities and their derivatives with respect to the Bernoulli parameters are expressed as power series in the load offered to the system, and a complete computational scheme to calculate the coefficients of these power series is derived. The derivation proceeds along the same lines as discussed in the previous chapter.

3.3.1 Balance equations

To apply the PSA to the present model, we first describe the system as a continuous-time Markov chain. Let $\boldsymbol{N}(t) = (N_1(t), \ldots, N_s(t))$ be the joint queue length at time t, $t \geq 0$. To transform the queue-length process into a Markov process, it is most appropriate to introduce a triple $(H(t), Z(t), \Xi(t))$ of supplementary variables, $t \geq 0$. The variable $H(t)$ will indicate the index of the queue that is being visited by the server (it changes at instants at which the server *leaves* a queue); the variable $Z(t)$ will indicate whether the server is switching ($Z(t) = 0$) or serving ($Z(t) = 1$); the variable $\Xi(t)$ will indicate the actual phase of either the current switch-over time or the current service time. Denote by $(\boldsymbol{N}, H, Z, \Xi)$ the joint vector of random variables with as distribution the stationary distribution of $(\boldsymbol{N}(t), H(t), Z(t), \Xi(t))$. For simplicity of the

discussion, it will be assumed throughout that the supplementary space is the same for all $\boldsymbol{n} \in \mathbb{N}^s$, and is given by

$$\mathcal{S} := \left\{(h,\zeta,\xi) \mid \ h = 1,\ldots,s,\ \ \zeta = 0,1,\ \ \xi = 1,\ldots,\Psi_h^\zeta\right\}, \tag{3.3}$$

while the states $(\boldsymbol{n},h,1,\xi)$ with $n_h = 0$ can not be entered (cf. also (3.7) below). The state probabilities are defined as follows: for $(\boldsymbol{n},h,\zeta,\xi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n},h,\zeta,\xi) := \Pr\{(\boldsymbol{N},H,Z,\Xi) = (\boldsymbol{n},h,\zeta,\xi)\}. \tag{3.4}$$

Equating the total rate out of each state to the total rate into that state yields the following set of global balance equations (cf. (2.3)).

For the states in which the server is switching, we have: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$,

$$\begin{aligned}
&\left[\chi\sum_{i=1}^{s} a_i + \mu_h^{0,\xi}\right] p(\boldsymbol{n},h,0,\xi) = \mu_h^{0,\xi+1} p(\boldsymbol{n},h,0,\xi+1) I\{\xi < \Psi_h^0\} \\
&+ \chi\sum_{i=1}^{s} a_i p(\boldsymbol{n}-\boldsymbol{e}_i,h,0,\xi) I\{n_i > 0\} \\
&+ \mu_{h-1}^{0,1}\pi_h^{0,\xi} p(\boldsymbol{n},h-1,0,1) I\{n_{h-1} = 0\} \\
&+ \mu_{h-1}^{1,1}\pi_h^{0,\xi} p(\boldsymbol{n}+\boldsymbol{e}_{h-1},h-1,1,1)\left[1 - q_{h-1} I\{n_{h-1} > 0\}\right].
\end{aligned} \tag{3.5}$$

The first term at the right-hand side of (3.5) indicates a phase transition in a switch-over time from Q_{h-1} to Q_h. The second term describes an arrival of a customer while the server is switching from Q_{h-1} to Q_h. The third term indicates that the server skips Q_{h-1} if this queue is empty and proceeds to Q_h immediately. The fourth term corresponds to a service completion at Q_{h-1} followed by a departure of the server from that queue.

For the states in which the server is serving, we have the following balance equations: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^1$, $n_h > 0$,

$$\begin{aligned}
&\left[\chi\sum_{i=1}^{s} a_i + \mu_h^{1,\xi}\right] p(\boldsymbol{n},h,1,\xi) = \mu_h^{1,\xi+1} p(\boldsymbol{n},h,1,\xi+1) I\{\xi < \Psi_h^1\} \\
&+ \chi\sum_{i=1}^{s} a_i p(\boldsymbol{n}-\boldsymbol{e}_i,h,1,\xi) I\{n_i > 0\} + \mu_h^{0,1}\pi_h^{1,\xi} p(\boldsymbol{n},h,0,1) \\
&+ q_h \mu_h^{1,1}\pi_h^{1,\xi} p(\boldsymbol{n}+\boldsymbol{e}_h,h,1,1).
\end{aligned} \tag{3.6}$$

The terms at the right-hand side of (3.6) can be interpreted as follows. The first term indicates a phase transition in a service time of a customer at Q_h. The second term corresponds to a customer arrival during the service of a customer at Q_h. The third term indicates an arrival of the server at Q_h, followed by an immediate service initiation at that queue. The last term corresponds to a service completion at Q_h and a subsequent service initiation of another customer at that queue.

Because the server can not be serving at a queue which is empty, we have: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^1$,

$$p(\boldsymbol{n}, h, 1, \xi) = 0, \text{ if } n_h = 0. \tag{3.7}$$

Further, according to the law of the total probability,

$$\sum_{(\boldsymbol{n},h,\zeta,\xi)\in\mathbb{N}^s\times\mathcal{S}} p(\boldsymbol{n}, h, \zeta, \xi) = 1. \tag{3.8}$$

We will also consider models in which some or all of the switch-over times vanish. When some of the switch-over times are 0 a.s., some straightforward modifications of the balance equations (3.5), (3.6) have to be made. When all switch-over times are equal to zero, there exists a unique zero state and the balance equations between the unique empty state and states with one customer present in the system have to be slightly modified (cf. [18] for the case of exponential service times and switch-over times).
It is readily verified that the process, conditioned on the event $\{\boldsymbol{N} = \boldsymbol{0}\}$, the $\boldsymbol{0}$-process, is irreducible. This is because the server keeps on moving along all queues when the system is empty. As discussed in section 2.3.2, the irreducibility of this $\boldsymbol{0}$-process is a sufficient condition for application of the PSA.

3.3.2 Computational scheme

In this section we discuss the use of the PSA for the present model. We follow the same lines as in section 2.3.3. The state probabilities and their derivatives with respect to the Bernoulli parameters are expressed as power series. We derive a complete recursive scheme to compute all coefficients for the state probabilities and their derivatives.

As discussed in section 2.3.3, the computed performance measures depend on the arrival rates a_i and the variable χ only through their products $\lambda_i = a_i\chi$, $i = 1, \ldots, s$. Because we want to compute derivatives with respect to the Bernoulli parameters q_i, $i = 1, \ldots, s$, the arrival rates may be normalized in such a way that $\chi = \rho$, the offered load to the system. This is convenient because ρ does not depend on the Bernoulli parameters (cf. the discussion in section 2.4.1).
Based on the basic property $p(\boldsymbol{n}, h, \zeta, \xi) = O(\rho^{|\boldsymbol{n}|})$, for $\rho \downarrow 0$, discussed in section 2.3.2 (cf. (2.5)), we introduce the following power-series expansions for the state probabilities: for $(\boldsymbol{n}, h, \zeta, \xi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, h, \zeta, \xi) = \rho^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \rho^k b_0(k; \boldsymbol{n}, h, \zeta, \xi). \tag{3.9}$$

Then the derivatives of the state probabilities can be expressed as power series in ρ as follows: for $(\boldsymbol{n}, h, \zeta, \xi) \in \mathbb{N}^s \times \mathcal{S}$, $r = 1, \ldots, s$,

$$\frac{\partial}{\partial q_r} p(\boldsymbol{n}, h, \zeta, \xi) = \rho^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \rho^k b_r(k; \boldsymbol{n}, h, \zeta, \xi), \tag{3.10}$$

where

$$b_r(k;\boldsymbol{n},h,\zeta,\xi) = \frac{\partial}{\partial q_r} b_0(k;\boldsymbol{n},h,\zeta,\xi). \tag{3.11}$$

Equation (3.11) follows from the fact that the variable of the power series expansions of the state probabilities, ρ, does not depend on the Bernoulli vector $\boldsymbol{q}$. Substituting these power-series expansions into the global balance equations (3.5) and (3.6), respectively, and equating the corresponding coefficients of powers of ρ, leads to the following set of equations for the coefficients: for $r = 0,1,\ldots,s$, $k = 0,1,\ldots,\quad \boldsymbol{n} \in \mathbb{N}^s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$,

$$\begin{aligned}
&\mu_h^{0,\xi} b_r(k;\boldsymbol{n},h,0,\xi) = \mu_h^{0,\xi+1} b_r(k;\boldsymbol{n},h,0,\xi+1) I\{\xi < \Psi_h^0\} \\
&+ \sum_{i=1}^{s} a_i b_r(k;\boldsymbol{n}-\boldsymbol{e}_i,h,0,\xi) I\{n_i > 0\} \\
&- \sum_{i=1}^{s} a_i b_r(k-1;\boldsymbol{n},h,0,\xi) I\{k > 0\} \\
&+ \mu_{h-1}^{0,1} \pi_h^{0,\xi} b_r(k;\boldsymbol{n},h-1,0,1) I\{n_{h-1} = 0\} \\
&+ \mu_{h-1}^{1,1} \pi_h^{0,\xi} b_r(k-1;\boldsymbol{n}+\boldsymbol{e}_{h-1},h-1,1,1) \\
&\qquad \times [1 - q_{h-1} I\{n_{h-1} > 0\}] I\{k > 0\} \\
&- \mu_{h-1}^{1,1} \pi_h^{0,\xi} b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_{h-1},h-1,1,1) \\
&\qquad \times I\{r = h-1\} I\{n_{h-1} > 0\} I\{k > 0\}
\end{aligned} \tag{3.12}$$

and for $r = 0,1,\ldots,s$, $k = 0,1,\ldots,\quad \boldsymbol{n} \in \mathbb{N}^s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^1$,

$$\begin{aligned}
&\mu_h^{1,\xi} b_r(k;\boldsymbol{n},h,1,\xi) = \mu_h^{1,\xi+1} b_r(k;\boldsymbol{n},h,1,\xi+1) I\{\xi < \Psi_h^1\} \\
&+ \sum_{i=1}^{s} a_i b_r(k;\boldsymbol{n}-\boldsymbol{e}_i,h,1,\xi) I\{n_i > 0\} \\
&- \sum_{i=1}^{s} a_i b_r(k-1;\boldsymbol{n},h,1,\xi) I\{k > 0\} \\
&+ \mu_h^{0,1} \pi_h^{1,\xi} b_r(k;\boldsymbol{n},h,0,1) \\
&+ q_h \mu_h^{1,1} \pi_h^{1,\xi} b_r(k-1;\boldsymbol{n}+\boldsymbol{e}_h,h,1,1) I\{k > 0\} \\
&+ \mu_h^{1,1} \pi_h^{1,\xi} b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_h,h,1,1) I\{k > 0\} I\{r = h\}.
\end{aligned} \tag{3.13}$$

To define an ordering of the states $(k;\boldsymbol{n},h,\zeta,\xi)$, we adopt the (partial) ordering $\prec$ over the $(k;\boldsymbol{n})$-combinations from (2.12): for $(k;\boldsymbol{n},h,\zeta,\xi)$, $(\hat{k};\hat{\boldsymbol{n}},\hat{h},\hat{\zeta},\hat{\xi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$\begin{aligned}
&(k;\boldsymbol{n},h,\zeta,\xi) \prec (\hat{k};\hat{\boldsymbol{n}},\hat{h},\hat{\zeta},\hat{\xi}) \\
&\text{if } \left[k + |\boldsymbol{n}| < \hat{k} + |\hat{\boldsymbol{n}}|\right] \vee \left[k + |\boldsymbol{n}| = \hat{k} + |\hat{\boldsymbol{n}}| \wedge k < \hat{k}\right].
\end{aligned} \tag{3.14}$$

To define an ordering of the triples $(h,\zeta,\xi) \in \mathcal{S}$, for given $(k;\boldsymbol{n}) \in \mathbb{N}^{1+s}$, a distinction has to be made between the empty and non-empty states.
Let us first consider the non-empty states. For given $(k;\boldsymbol{n})$, $\boldsymbol{n} \neq \boldsymbol{0}$, *and* $h \in \{1,\ldots,s\}$, we define the following ordering: for $(k;\boldsymbol{n},h,\zeta,\xi)$, $(k;\boldsymbol{n},h,\hat{\zeta},\hat{\xi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$(k;\boldsymbol{n},h,\zeta,\xi) \prec (k;\boldsymbol{n},h,\hat{\zeta},\hat{\xi}) \text{ if } \left[\zeta < \hat{\zeta}\right] \vee \left[\zeta = \hat{\zeta} \wedge \xi > \hat{\xi}\right], \quad (3.15)$$

so that the triples (h,ζ,ξ) are ranked in increasing order as

$$(h,0,\Psi_h^0),\ (h,0,\Psi_h^0-1),\ldots,(h,0,1),(h,1,\Psi_h^1),\ldots,(h,1,1). \quad (3.16)$$

For $\boldsymbol{n} \neq \boldsymbol{0}$, it remains to define an ordering over the values of $h \in \{1,\ldots,s\}$. This ordering will generally depend on $\boldsymbol{n}$. Because $\boldsymbol{n} \neq \boldsymbol{0}$, there exists an $h^*_{\boldsymbol{n}} \in \{1,\ldots,s\}$ such that $n_{h^*_{\boldsymbol{n}}-1} > 0$. Define the following ordering: for $(k;\boldsymbol{n},h,\zeta,\xi),\ (k;\boldsymbol{n},\hat{h},\hat{\zeta},\hat{\xi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$\begin{aligned} &(k;\boldsymbol{n},h,\zeta,\xi) \prec (k;\boldsymbol{n},\hat{h},\hat{\zeta},\hat{\xi}) \\ &\text{if } (h-h^*_{\boldsymbol{n}}) \bmod s \ < \ (\hat{h}-h^*_{\boldsymbol{n}}) \bmod s, \end{aligned} \quad (3.17)$$

so that the triples (h,ζ,ξ) are ranked in increasing order with respect to h as $h^*_{\boldsymbol{n}}, h^*_{\boldsymbol{n}}+1,\ldots,s,1,\ldots,h^*_{\boldsymbol{n}}-1$. Thus, for given $(k;\boldsymbol{n})$ the triples (h,ζ,ξ) and $(\hat{h},\hat{\zeta},\hat{\xi})$ are lexicographically ordered according to (3.17), for $h \neq \hat{h}$, and to (3.15) for $h = \hat{h}$.

Under the ordering $\prec$ of the states $(k;\boldsymbol{n},h,\zeta,\xi)$, defined by (3.14), (3.15), and (3.17), the set of equations (3.12), (3.13) forms a recursive scheme for the coefficients $b_0(k;\boldsymbol{n},h,\zeta,\xi)$ (except for the states with $\boldsymbol{n} = \boldsymbol{0}$). Note that this ordering is partial, so that it leaves some degree of freedom in the order in which the terms are computed. In order to derive a computational scheme for the coefficients $b_r(k;\boldsymbol{n},h,\zeta,\xi)$, $r = 0,1,\ldots,s$, we extend the partial ordering $\prec$ to the following partial ordering $\tilde{\prec}$ of the vectors $(r,k;\boldsymbol{n},h,\zeta,\xi) \in \{0,1,\ldots,s\} \times \mathbb{N}^{1+s} \times \mathcal{S}$:

$$\begin{aligned} &(r,k;\boldsymbol{n},h,\zeta,\xi) \ \tilde{\prec} \ (\hat{r},\hat{k};\hat{\boldsymbol{n}},\hat{h},\hat{\zeta},\hat{\xi}) \\ &\text{if } \ [r=0 \wedge \hat{r}>0] \vee \left[r=\hat{r} \wedge (k;\boldsymbol{n},h,\zeta,\xi) \prec (\hat{k};\hat{\boldsymbol{n}},\hat{h},\hat{\zeta},\hat{\xi})\right]. \end{aligned} \quad (3.18)$$

Under the ordering $\tilde{\prec}$ the set of equations (3.12), (3.13) forms a recursive scheme for all coefficients $b_r(k;\boldsymbol{n},h,\zeta,\xi)$, for $(k;\boldsymbol{n},h,\zeta,\xi) \in \mathbb{N}^{1+s} \times \mathcal{S}$, $r = 0,1,\ldots,s$ (except for those states with $\boldsymbol{n} = \boldsymbol{0}$). This is because equations (3.12) and (3.13) express the coefficients $b_r(k;\boldsymbol{n},h,\zeta,\xi)$ in terms of coefficients of lower order with respect to $\tilde{\prec}$.

Hence, the only states that require further attention are the states with $\boldsymbol{n} = \boldsymbol{0}$ and hence, from (3.7), $\zeta = 0$. For these states, equations (3.12) read: for $r = 0,1,\ldots,s,\ k = 0,1,\ldots,\ h = 1,\ldots,s,\ \xi = 1,\ldots,\Psi_h^0$,

$$\begin{aligned} \mu_h^{0,\xi} b_r(k;\boldsymbol{0},h,0,\xi) \ &= \ \mu_h^{0,\xi+1} b_r(k;\boldsymbol{0},h,0,\xi+1) I\{\xi < \Psi_h^0\} \\ &+ \ \mu_{h-1}^{0,1} \pi_h^{0,\xi} b_r(k;\boldsymbol{0},h-1,0,1) + y_r(k;h,\xi). \end{aligned} \quad (3.19)$$

The quantities $y_r(k;h,\xi)$, $r = 0,1,\ldots,s,\ h = 1,\ldots,s,\ \xi = 1,\ldots,\Psi_h^0$, are defined by $y_r(0;h,\xi) := 0$ and for $k = 1,2,\ldots,$ by

$$\begin{aligned} y_r(k;h,\xi) &:= \mu_{h-1}^{1,1}\pi_h^{0,\xi} b_r(k-1;\boldsymbol{e}_{h-1},h-1,1,1) \\ &\quad - \sum_{i=1}^{s} a_i b_r(k-1;\boldsymbol{0},h,0,\xi). \end{aligned} \tag{3.20}$$

Note that the quantities $y_r(k;h,\xi)$ consist of terms with coefficients of lower order with respect to $\tilde{\prec}$ than $(r,k;\boldsymbol{0},h,0,\xi)$ and hence, can be considered to be known in (3.19). For k and r fixed, the sets of equations (3.19) are dependent. However, from the law of total probability (3.8) and the power-series expansions (3.9), we have: for $r = 0,1,\ldots,s$,

$$\sum_{h=1}^{s}\sum_{\xi=1}^{\Psi_h^0} b_r(0;\boldsymbol{0},h,0,\xi) = I\{r=0\}, \tag{3.21}$$

and for $k = 1,2,\ldots,$

$$\sum_{h=1}^{s}\sum_{\xi=1}^{\Psi_h^0} b_r(k;\boldsymbol{0},h,0,\xi) = -Y_r(k), \text{ with} \tag{3.22}$$

$$Y_r(k) := \sum_{0<|\boldsymbol{n}|\le k}\ \sum_{(h,\zeta,\xi)\in\mathcal{S}} b_r(k-|\boldsymbol{n}|;\boldsymbol{n},h,\zeta,\xi). \tag{3.23}$$

Consider, for k fixed, the set of equations consisting of all but one of the equations (3.19) combined with either (3.21) or (3.22). Then the determinant of these sets of equations is given by $\Delta := \sigma_1 \prod_{h=1}^{s}\prod_{\xi=1}^{\Psi_h^0}\mu_h^{0,\xi} > 0$, independent of k, so that the set of equations (3.19) combined with either (3.21) or (3.22) is indeed uniquely solvable for each k. For $k = 0$, this set of equations is readily solved: for $r = 0,1,\ldots,s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$,

$$b_r(0;\boldsymbol{0},h,0,\xi) = \frac{1}{\sigma_1\mu_h^{0,\xi}}\left(\sum_{\psi=\xi}^{\Psi_h^0}\pi_h^{0,\psi}\right) I\{r=0\}. \tag{3.24}$$

It is tedious but straightforward to show that for $r = 0,1,\ldots,s$, $k = 1,2,\ldots,$

$$\begin{aligned} &b_r(k;\boldsymbol{0},s,0,1) = \\ &\tfrac{-1}{\sigma_1\mu_s^{0,1}} \times \left[Y_r(k) + \sum_{h=1}^{s}\sum_{\xi=1}^{\Phi_h^0}\tfrac{1}{\mu_h^{0,\xi}} \right. \\ &\qquad \left. \times \sum_{\psi=\xi}^{\Phi_h^0}\left\{ y_r(k;h,\psi) + \pi_h^{0,\psi}\sum_{j=1}^{h-1}\sum_{\nu=1}^{\Psi_j^0} y_r(k;j,\nu)\right\}\right]. \end{aligned} \tag{3.25}$$

Summarizing, the coefficients $b_r(k;\boldsymbol{n},h,\zeta,\xi)$, $(k;\boldsymbol{n},h,\zeta,\xi) \in \mathbb{N}^{1+s}\times\mathcal{S}$, $r = 0,1,\ldots,s$, can be recursively computed (up to the, say, M-th power of ρ) using the following computational scheme:

step 1 : determine $b_r(0;\boldsymbol{0},h,0,\xi)$, $r = 0,1,\ldots,s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$, according to (3.24);

step 2 : m:=1;

step 3 : for all $(k;\boldsymbol{n},h,\zeta,\xi) \in \mathbb{N}^{1+s} \times \mathcal{S}$, with $(\boldsymbol{n},\zeta) \neq (\boldsymbol{0},0)$ and with $k+|\boldsymbol{n}| = m$, determine $b_r(k;\boldsymbol{n},h,\zeta,\xi)$, $r = 0,1,\ldots,s$, according to (3.12) and (3.13) (in increasing order of $(r,k;\boldsymbol{n},h,\zeta,\xi)$ with respect to $\tilde{\prec}$);

step 4 : determine $b_r(m;\boldsymbol{0},s,0,1), r = 0,1,\ldots,s$, according to (3.25), and update the values of $f^{(l)}(m)$ and $f_r^{(l)}(m)$, $r = 1,\ldots,s$, $l = 1,\ldots,L$;

step 5 : compute $b_r(m;\boldsymbol{0},h,0,\xi)$, $r = 0,1,\ldots,s$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$, according to (3.19); the order of the couples (h,ξ) is $(1,\Psi_h^0),\ldots,(1,1)$, $\ldots,(s,\Psi_h^0),\ldots,(s,2)$;

step 6 : $m := m+1$; if $m \leq M$ then return to *step 3* ; otherwise STOP.

In (2.18), (2.19), (2.48) and (2.49) we expressed a general type of performance measures, and their derivatives with respect to the control variable γ, in terms of the coefficients of the power series of the state probabilities. For the present model, the formal supplementary variable $\boldsymbol{\Phi}$ and the corresponding realization $\boldsymbol{\varphi}$ should be replaced by the triples (H,Z,Ξ) and (h,ζ,ξ), respectively. Then, general performance measures of the form $\mathrm{E}\{g^{(l)}(\boldsymbol{n},h,\zeta,\xi)\}$, $l = 1,\ldots,L$, and their derivatives with respect to the Bernoulli parameters, can be determined by means of the PSA along the same lines as discussed in sections 2.3.3 and 2.4.1. The mean queue length at Q_i (including the customer in service) can be determined by taking $g^{(i)}(\boldsymbol{n},h,\zeta,\xi) := n_i$ in (2.19), $i = 1,\ldots,s$. Once the mean queue lengths have been determined, the mean waiting times $\mathrm{E}W_i$, $i = 1\ldots,s$, can be obtained directly from Little's formula.

In chapter 2 we presented the use of the PSA for general QBD processes. Although the states $(k;\boldsymbol{n},\boldsymbol{\varphi})$ could be ordered mainly recursively, for each $(k;\boldsymbol{n})$-combination a set of $|\mathcal{S}|$ linear equations would have to be solved. In the present chapter this general approach has been worked out for polling systems with Bernoulli schedules and Coxian distributed service times and switch-over times. We have derived a *fully recursive* computation order by exploring the specific structure of the model (cf. (3.15), (3.17)).
In the derivation of the computational scheme we explicitly used the fact that the service times and switch-over times are Coxian distributed, rather than according to some more general phase-type distribution. By using Coxian distributions an ordering of the supplementary variables ξ for given $(k;\boldsymbol{n},h,\zeta)$ (cf. (3.15)) can be defined such that the coefficients $b_r(k;\boldsymbol{n},h,\zeta,\xi)$, $\xi = \Psi_h^\zeta,\ldots,1$, can be computed recursively. This is because for Coxian distributions the phases can be ordered in such a way that phase transitions can only occur to phases with a lower index, as opposed to more general phase-type distributions in which phase transitions may occur from each phase to each other phase. Therefore, Coxian distributions are commonly preferred to more general phase-type distributions (cf. also the remarks in section 2.5).

For models in which all switch-over times are zero, the presence of a unique zero state leads to a different set of balance equations, which differs from the balance equations discussed here in the balance between the empty state and states with one customers in the system. The computational scheme is somewhat simplified due to the presence of a unique zero state. The equations (3.19) between zero states vanish. The first term of the power-series expansions of the unique zero state, say $b_r(0;\mathbf{0})$, $r = 0, 1, \ldots, s$, follows directly from the law of total probability (3.8) and is explicitly given by $I\{r = 0\}$ (cf. (3.24)). We refer to [18] for a detailed discussion for the case of exponentially distributed service times. The computational scheme for the partial derivatives can be obtained in a similar way as has been done in this section.

The computational scheme derived in this section can be readily extended to polling systems in which the server is routing along the queues according to a fixed service order table (cf. [110]). We refer to [25] for a more detailed discussion on the use of the PSA for general polling tables with the extension to the computation of derivatives.

Higher-order derivatives with respect to the Bernoulli parameters can, in principle, be computed by means of the PSA following the same approach as discussed here for first order derivatives.

Among the few general exact results that have been obtained for polling systems are the formulations of the PCLs (cf. [40]). For a cyclic polling system with Bernoulli schedules at all queues the PCL reads (cf. [162], [38]):

$$\begin{aligned} &\sum_{i=1}^{s} \rho_i \left[1 - a_i(1-q_i)\tfrac{\sigma_1\rho}{1-\rho}\right] \mathrm{E}W_i = \\ &\tfrac{\rho^2}{1-\rho}\tfrac{\beta_2}{2\beta_1} + \rho\tfrac{\sigma_2}{2\sigma_1} + \tfrac{\sigma_1}{1-\rho}\sum_{i=1}^{s}\rho_i^2(1-q_i) + \tfrac{\sigma_1}{2(1-\rho)}\left[\rho^2 - \sum_{i=1}^{s}\rho_i^2\right]. \end{aligned} \tag{3.26}$$

The PCL can be used to give an indication of the accuracy of the computation with the PSA. The accuracy of the computed mean waiting times can be roughly estimated by determining the left-hand side of (3.26) on the basis of the computed mean waiting times and comparing this value to the exact value of the right-hand side of (3.26). We emphasize that the PCL can only give a rough indication of the accuracy of the computation, because the computation error in each of the individual mean waiting times may still be considerably larger.

3.4 Properties of optimal Bernoulli schedules

In this section we discuss some properties of optimal combinations of Bernoulli parameters. In sections 3.4.1 and 3.4.2 we derive light- and heavy-traffic limits

of optimal Bernoulli schedules. In section 3.4.3 we give a partial solution to the optimization problem, based on the $c\mu$-rule for priority systems.

3.4.1 Light-traffic properties

An elegant way to get an insight into the light-traffic behavior of the system is the use of the PSA, as elaborated in the previous section. The algorithm relies on the property that the state probabilities can be expressed as power series in ρ. The coefficients of these state probabilities, and of the mean queue lengths, can be determined according to the computational scheme discussed in section 3.3.2. Useful information about the optimization of Bernoulli schedules in light-traffic systems can be obtained by determining the first few terms of the power-series expansions of the mean waiting times in a *symbolic* manner. The symbolic computation of the first few terms of the power-series expansions of the mean waiting times according to the computational schemes is tedious, but straightforward. To this end, we used an algebraic formula manipulation computer program. Because the PSA is applicable to systems that can be modeled as QBD processes, general probability distributions for the service times and the switch-over times are approximated by Coxian distributions. However, the capacity of the symbolic manipulator has restricted the computation to models with 3 queues, using 2-phase Coxian distributions. The computations suggest the following light-traffic asymptotes of the mean waiting times:

if $\sigma_1 = 0$, then for $i = 1, \ldots, s$,

$$\mathrm{E}W_i = \rho\frac{\beta_2}{2\beta_1} + \rho^2\left[\eta_i\frac{\beta_2}{2\beta_1} + \Upsilon(i) + \Gamma(i,\boldsymbol{q})\right] + O(\rho^3), \quad \rho \downarrow 0, \tag{3.27}$$

with

$$\Upsilon(i) := \sum_{j=1}^{s-1}\eta_{i+j}\sum_{k=0}^{j-1}\eta_{i+k}\frac{\beta_{i+k}^{(2)}}{\beta_{i+k}^{(1)}}, \quad \Gamma(i,\boldsymbol{q}) := \sum_{j=1}^{s} q_j\eta_j^2\frac{\beta_j^{(2)}}{\beta_j^{(1)}} - q_i\eta_i\frac{\beta_i^{(2)}}{\beta_i^{(1)}}; \tag{3.28}$$

if $\sigma_1 > 0$, then for $i = 1, \ldots, s$,

$$\begin{aligned}\mathrm{E}W_i &= \tfrac{\sigma_2}{2\sigma_1} \\ &+ \rho\left[\tfrac{\beta_2}{2\beta_1}(\sigma_1 - \tfrac{\sigma_2}{2\sigma_1})(1-\eta_i) + \aleph(i) + \Omega(i,\boldsymbol{q})\right] + O(\rho^2), \ \rho \downarrow 0,\end{aligned} \tag{3.29}$$

with

$$\aleph(i) := \sum_{j\neq i}\eta_j\sum_{k=i+1}^{j}\frac{\sigma_k^{(2)} - \sigma_k^{(1)^2}}{\sigma_1}, \quad \Omega(i,\boldsymbol{q}) := (1-q_i)(\eta_i\sigma_1 + \frac{1}{2}a_i\sigma_2). \tag{3.30}$$

The validity of the coefficients in (3.27) and (3.29) has been checked numerically for numerous examples with more than 3 queues and more than 2 phases, and was found to be valid in all considered cases. Moreover, one may verify that the coefficients in (3.27) and (3.29) are in agreement with the PCL (3.26).

The coefficient of ρ^2 in (3.27) consists of three parts. The term $\Upsilon(\cdot)$ reflects the influence of the order in which the server visits the queues and the term $\Gamma(\cdot,\cdot)$ depends on the service discipline at the queues.
The coefficient of ρ in (3.29) consists of four parts. The term $\aleph(\cdot)$ depends on the individual switch-over time distributions and is independent of the service disciplines. The term $\Omega(\cdot,\cdot)$ depends on the switch-over time distributions only through the first two moments of the total switch-over time distribution per cycle, and it depends on the service disciplines at the queues.

For models with zero and non-zero switch-over times, light-traffic asymptotes of $\boldsymbol{q}^*$ can be derived from (3.27) and (3.29), respectively.
Let us first consider the case $\sigma_1 = 0$. Omitting the terms that do not depend on $\boldsymbol{q}$ in (3.27), it follows that the coefficient of ρ^2 of the cost function (3.2) is $\Sigma_{i=1}^{s} q_i\eta_i(\eta_i - c_i)\beta_i^{(2)}/\beta_i^{(1)}$, yielding the following light-traffic asymptotes of $\boldsymbol{q}^*$: for $i = 1,\ldots,s$,

$$\text{(1) if } \sigma_1 = 0 \text{ and } c_i > \eta_i, \text{ then } \lim_{\rho\downarrow 0} q_i^* = 1; \tag{3.31}$$

$$\text{(2) if } \sigma_1 = 0 \text{ and } c_i < \eta_i, \text{ then } \lim_{\rho\downarrow 0} q_i^* = 0. \tag{3.32}$$

The case $c_i = \eta_i$ is not covered by (3.31) and (3.32). Apparently, the light-traffic limit of $\boldsymbol{q}^*$ is in this case determined by higher-order terms of the power-series expansions (3.27).

Let us next consider the case $\sigma_1 > 0$. Then (3.29) yields the following light-traffic limit of $\boldsymbol{q}^*$:

$$\text{(3) if } \sigma_1 > 0 \text{ then } \lim_{\rho\downarrow 0} q_i^* = 1, \ i = 1,\ldots,s. \tag{3.33}$$

The limits in (3.31)-(3.33) are defined such that the total arrival rate tends to zero, while the *ratios* between the individual arrival rates remain fixed. Note that if σ_1 becomes very small, the difference in the cost associated with $q_i = 0$ and $q_i = 1$ tends to zero, in light traffic.

For small values of ρ it follows directly from (3.9), (3.10) and (3.11) that the state probabilities and hence, the performance measures, are (partially) differentiable with respect to the Bernoulli parameter $\boldsymbol{q}$. It is assumed that the performance measures are also differentiable for larger values of ρ.
Because the light-traffic limits of q_i^* are boundary values, the light-traffic limits (3.31), (3.32) and (3.33) not only hold in the limiting case $\rho \downarrow 0$, but remain valid when $0 \leq \rho < \epsilon$, for some ϵ small enough. More precisely, cases were found with $\sigma_1 > 0$ small and $c_i < \eta_i$, where there exist positive numbers $\rho^{(0)}$, $\rho^{(1)}$ and $\rho^{(2)}$, satisfying $0 < \rho^{(1)} < \rho^{(2)} < \rho^{(0)}$, such that $q_i^* = 1$ for $0 \leq \rho \leq \rho^{(1)}$, q_i^* decreases from one to zero as ρ increases from $\rho^{(1)}$ to $\rho^{(2)}$ and $q_i^* = 0$ for $\rho^{(2)} \leq \rho \leq \rho^{(0)}$, cf. e.g. Figure 3.1 in section 3.5.

The fact that $\boldsymbol{q}$ appears in the ρ-term in (3.29) and does *not* appear in the ρ-term in (3.27) can be explained as follows. The ρ^k-terms in (3.27) and (3.29) correspond to states of the system in which at most k customers are present in the system, and using PASTA (cf. [173]), to situations in which an arriving customer finds at most k customers present in the system upon arrival, $k = 0, 1, \ldots$. Consider a marked customer C_A arriving at Q_i, while there is only one customer C_B present in the system, which is residing at Q_i. Moreover, suppose that there are no arrivals during the waiting time of C_A (higher-order effect). Then the waiting time of C_A is equal to the residual sojourn time of C_B *plus*, with probability 1-q_i, a complete cycle of the server along the queues. Hence, if $\sigma_1 > 0$, the waiting time of C_A clearly depends on q_i. If $\sigma_1 = 0$, the extra cycle time vanishes, because of the assumption that no other customers arrive during the waiting time of C_A and hence, the waiting time of C_A does *not* depend on q_i $(i = 1, \ldots, s)$.

3.4.2 Heavy-traffic properties

In this section we first discuss some properties of the system in heavy traffic. Then, we derive heavy-traffic asymptotes for the optimal Bernoulli schedules. For notational convenience, define

$$\tau := \rho \left[1 + \sigma_1 \max_{i=1,\ldots,s} a_i(1 - q_i) \right], \tag{3.34}$$

and let

$$\mathcal{T} := \left\{ 1 \leq i \leq s \mid a_i(1 - q_i) = \max_{j=1,\ldots,s} \{a_j(1 - q_j)\} \right\}. \tag{3.35}$$

From the stability condition (3.1) it follows that the system tends to the boundary of the stability region when $\tau \uparrow 1$. It should be noted here that, according to definition (3.34), the variable τ may not be differentiable with respect to the control parameter $\boldsymbol{q}$. However, this difficulty may be overcome for *given* $\boldsymbol{q} = \boldsymbol{q}^{(0)} = (q_1^{(0)}, \ldots, q_s^{(0)})$ by replacing q_i by $q_i^{(0)}$ in (3.34). Then, one may verify that τ is differentiable with respect to $\boldsymbol{q}$, while the boundary of the stability region is still reached when $\tau \uparrow 1$ (cf. the discussion at the end of section 2.4.1).

When the number of queues is not too large and the parameters of the system are not too asymmetrical, it is possible to obtain accurate data for performance measures even for high occupancy of the system (i.e. τ close to one) by means of the PSA. It is shown by Fricker and Jaïbi [85] that not all queues become instable when $\tau \uparrow 1$. They state that Q_i becomes instable for $\tau \uparrow 1$ if and only if $i \in \mathcal{T}$. However, they do not make any statements about the *order* of the poles of the mean waiting times at $\tau = 1$. Numerous numerical experiments have confirmed the following observation concerning the poles of the individual mean waiting times at $\tau = 1$:

$$\begin{aligned} &\text{if } i \in \mathcal{T}, \text{ then } EW_i \text{ has a pole of order 1 at } \tau = 1; \\ &\text{otherwise, } EW_i \text{ has a finite limit for } \tau \uparrow 1. \end{aligned} \tag{3.36}$$

And for the derivatives of the mean waiting times with respect to the Bernoulli parameters, we have found from numerical experiments:

$$\begin{aligned} &\text{if } (\sigma_1 = 0 \text{ and } |\mathcal{T}| \geq 2) \text{ or if } \sigma_1 > 0, \text{ then if } i \in \mathcal{T} \text{ and } r \in \mathcal{T}, \\ &\quad \text{then } \tfrac{\partial EW_i}{\partial q_r} \text{ has a pole of order 2 at } \tau = 1; \\ &\text{in all other cases, } \tfrac{\partial EW_i}{\partial q_r} \text{ has a finite limit for } \tau \uparrow 1. \end{aligned} \tag{3.37}$$

In practice, these properties may be very useful for accelerating the convergence of the epsilon algorithm, as elaborated upon in section 2.4.3.

Let us reconsider the heavy-traffic behavior of the mean waiting times. Denote the heavy-traffic *residue* of the mean waiting time at Q_i by

$$\omega_i := \lim_{\tau \uparrow 1} (1 - \tau) EW_i, \quad i = 1, \ldots, s. \tag{3.38}$$

Then, according to the heavy-traffic properties (3.36), $\omega_i > 0$ if $i \in \mathcal{T}$ and ω_i=0 if $i \notin \mathcal{T}$.
There is no closed-form expression known for ω_i, except for a few exceptional cases. As an example of such a special case, consider a model with exhaustive service at all queues, i.e. $q_i = 1$, $i = 1, \ldots, s$. For this model, one can obtain explicit expressions for heavy-traffic residues of the mean waiting times from the set of equations in chapter 4 in [158]: if $q_i = 1$, $i = 1, \ldots, s$, then

$$\omega_i = \frac{1 - \eta_i}{\sum_{j=1}^{s} \eta_j (1 - \eta_j)} \frac{\beta_2}{2\beta_1} + \frac{1}{2} \sigma_1 (1 - \eta_i), \quad i = 1, \ldots, s. \tag{3.39}$$

Moreover, if $\omega_i = 0$, then the finite limit for the mean waiting time at Q_i does not admit a closed-form expression, except for a few special cases, e.g. when $\mathcal{T} = \{1, \ldots, s\} \setminus \{i\}$.

Let us now discuss the heavy-traffic limit of the optimal Bernoulli schedule $\boldsymbol{q}^*$. If $\sigma_1 > 0$, the stability condition (3.1) can be reformulated as

$$q_i > 1 - \frac{1 - \rho}{\sigma_1 a_i \rho}, \quad i = 1, \ldots, s. \tag{3.40}$$

Hence, (3.40) induces a lower bound on the set of possible values of q_i, $i = 1, \ldots, s$. Moreover, as the right-hand side of (3.40) tends to one as $\rho \uparrow 1$, we have the following heavy-traffic asymptote for q_i^*:

$$\text{if } \sigma_1 > 0, \text{ then } \lim_{\rho \uparrow 1} q_i^* = 1, \quad i = 1, \ldots, s. \tag{3.41}$$

If $\sigma_1 = 0$, then the stability condition reads $\rho < 1$ and hence, $\boldsymbol{q}$ has no influence on the stability of the system. Thus, unlike in the case $\sigma_1 > 0$, the ergodicity condition (3.1) does *not* imply that q_i tends to 1, for $i = 1, \ldots, s$, as $\rho \uparrow 1$ (or equivalently, $\tau \uparrow 1$). In fact, cases have been found in which the heavy-traffic asymptote of $\boldsymbol{q}^*$ lies in the interior of [0,1], cf. e.g. Figure 3.2.

3.4.3 Partial solution

In this section we present a partial solution of the optimization problem. The solution is a simple index rule, giving explicit values of some of the components of the optimal Bernoulli schedule $\boldsymbol{q}^*$. It states that those queues i for which the ratio c_i/ρ_i is maximal over all queues should be served exhaustively. This (partly conjectured) index rule, which is supported by the $c\mu$-rule for priority systems, may be very useful because it is trivial to implement and because it may considerably decrease the dimension of the remaining problem to find the components of $\boldsymbol{q}^*$ which are not covered by this rule. Interestingly, these conjectured decision rules can be obtained alternatively from (also partly conjectured) monotonicity properties of the mean waiting times, enlarging the credibility of the rather intuitive solution.
First, we present the partial solution of the optimization problem considered in this chapter and give an intuitive explanation why this solution follows from the $c\mu$-rule for priority systems. Then, we present an alternative derivation of the partial solution. To this end, we present (partly conjectured) monotonicity properties and show that the partial solution follows directly from these properties.

The following *conjectured* decision rules partially solve the optimization problem: for $i = 1, \ldots, s$,

$$(1) \text{ if } \frac{c_i}{\rho_i} = \max_{j=1,\ldots,s} \frac{c_j}{\rho_j}, \text{ then } q_i^* = 1; \tag{3.42}$$

$$(2) \text{ if } \sigma_1 = 0, \text{ then if } \frac{c_i}{\rho_i} = \min_{j=1,\ldots,s} \frac{c_j}{\rho_j}, \text{ then } q_i^* = 0. \tag{3.43}$$

Decision rules (3.42) and (3.43) follow from the classical $c\mu$-rule for priority systems. For systems with zero switch-over times and in which the server can choose the next queue to be visited after each service completion of a customer, the $c\mu$-rule gives priorities to the queues in increasing order of the values of c_i/ρ_i. For these systems, the $c\mu$-rule can be shown to minimize a weighted sum of the mean waiting times (cf. [130], [58]). Hence, if a queue i, for which $c_i/\rho_i = \max_{j=1,\ldots,s} c_j/\rho_j$, is non-empty upon a service completion epoch at Q_i, then according to the $c\mu$-rule, it is not optimal for the server to proceed to another queue; and if the server has just completed a service at Q_k, for which $c_k/\rho_k = \min_{j=1,\ldots,s} c_j/\rho_j$, then according to the $c\mu$-rule, the server should depart from Q_k to check whether customers are waiting for service at other queues. When the switch-over times increase, the optimal values of the Bernoulli parameters tend to increase, because the server should serve more jobs at each visit to compensate for the loss of its availability due to the switches. This implies that when $\sigma_1 > 0$ a queue i with c_i/ρ_i maximal over all queues should still be served exhaustively, while a queue i with c_i/ρ_i minimal may require a positive q_i (cf. also (3.40)).

In the case $c_i = \rho_i/\rho$ $(i = 1, \ldots, s)$, the cost function $C(\boldsymbol{q})$ (cf. (3.2)) is proportional to the mean amount of waiting work in the system. The latter can be shown to be minimal when all queues are served exhaustively (cf. [123]), i.e. $q_i^* = 1$, $i = 1, \ldots, s$. Moreover, it follows from (3.26) that if $\sigma_1 = 0$ then $C(\boldsymbol{q})$ is independent of $\boldsymbol{q}$, so that every feasible value of $\boldsymbol{q}$ is optimal. These results are in agreement with the index rules (3.42) and (3.43).

Alternatively, the following combination of (partly conjectured) arguments also supports the validity of the partial solutions (3.42) and (3.43). First, it follows from results in [123] that:

$$\frac{\partial}{\partial q_i} \sum_{j=1}^{s} \rho_j \mathrm{E}W_j \leq 0, \quad i = 1, \ldots, s. \tag{3.44}$$

This observation is an immediate consequence of a stronger property (in the transient regime) proved by Levy et al. [123], who state that $V(t)$, the total amount of unfinished work in the system at time t, is stochastically non-increasing in q_i, $i = 1, \ldots, s$, for $t \geq 0$. As a consequence, the steady-state mean amount of unfinished work in the system, $\mathrm{E}V$, is also non-increasing in q_i, $i = 1, \ldots, s$. The monotonicity result (3.44) follows then immediately from the relation (cf. [94], [55]) $\mathrm{E}V = \Sigma_{j=1}^{s} \rho_j \mathrm{E}W_j + \Sigma_{j=1}^{s} \rho_j \beta_j^{(2)}/2\beta_j^{(1)}$.

The second observation (conjectured) is that the following monotonicity properties hold: for $i = 1, \ldots, s$,

$$\mathrm{E}W_i \text{ is decreasing in } q_i; \tag{3.45}$$

$$\mathrm{E}W_i \text{ is increasing in } q_j \text{ for all } j \neq i. \tag{3.46}$$

This conjecture (3.45), (3.46) has been checked for numerous numerical examples, and was found to be valid in all considered cases. Unfortunately, it appears to be very hard to prove this property rigorously. The following arguments support the validity of the conjecture. Let us consider the effect of the parameters q_j on $\mathrm{E}W_i$ $(j \neq i)$, and let us view services at Q_j as switch-over periods, whose duration is distributed as the convolution of random variables, whose number is the number of customers being served at Q_j. Denote by $\tilde{N}_j$ the number of customers served at Q_j during one visit of the server to that queue. Then simple balancing arguments imply that $\mathrm{E}\tilde{N}_j = \lambda_j \sigma_1/(1-\rho)$, *independent of* q_j. However, the second moment of $\tilde{N}_j$ *does* depend on q_j. Increasing the Bernoulli parameter q_j will increase $\tilde{N}_j$ at times Q_j happens to be relatively long and thus is likely to decrease $\tilde{N}_j$ when Q_j is relatively short. As a consequence, increasing the value of q_j is likely to lead to an increase of the variance in the number of customers served per visit at Q_j. From the viewpoint of Q_i, increasing q_j implies that the customers at Q_i observe switch-over periods of the same mean, but with higher variance. Considering Q_i as an isolated queue with server vacations (where the services of the other queues $j \neq i$ together

with the switch-over times constitute one large vacation), it is likely that the mean waiting time at Q_i increases when the second moment of the vacations is increased (while keeping their first moments the same). Combining this conjecture (3.46) with property (3.44) implies the validity of (3.45).

We will now show that combining (3.44) together with conjectures (3.45) and (3.46) imply the validity of decision rules (3.42) and (3.43). Suppose $c_i/\rho_i \geq c_j/\rho_j$, $j = 1, \ldots, s$. Then it follows that: for $i = 1, \ldots, s$,

$$\begin{aligned} \frac{\partial}{\partial q_i} \sum_{j=1}^{s} c_j \mathrm{E}W_j &= \frac{c_i}{\rho_i} \frac{\partial}{\partial q_i} \rho_i \mathrm{E}W_i + \sum_{j \neq i} \frac{c_j}{\rho_j} \frac{\partial}{\partial q_i} \rho_j \mathrm{E}W_j \\ &\leq \frac{c_i}{\rho_i} \frac{\partial}{\partial q_i} \rho_i \mathrm{E}W_i + \frac{c_i}{\rho_i} \sum_{j \neq i} \frac{\partial}{\partial q_i} \rho_j \mathrm{E}W_j \\ &= \frac{c_i}{\rho_i} \frac{\partial}{\partial q_i} \sum_{j=1}^{s} \rho_j \mathrm{E}W_j \leq 0. \end{aligned} \tag{3.47}$$

From (3.47) it follows that $q_i^* = 1$, which confirms the validity of (3.42).
To consider the validity of (3.43), suppose $c_i/\rho_i \leq c_j/\rho_j$, $j = 1, \ldots, s$, and let $\sigma_1 = 0$. Then the first inequality in (3.47) is reversed. Moreover, because $\sigma_1 = 0$ equality holds in (3.44). As a result, the left-hand side of (3.47) is non-negative, which confirms the validity of (3.43).

To the best of the author's knowledge, similar monotonicity properties of the mean waiting times for polling systems have not been found in the literature, except for a recent paper of Borst et al. [35]. They suggest similar monotonicity properties for polling systems with (fixed) limited service at all queues. They also point out that such a monotonicity seems to be difficult to prove. The above-mentioned intuitive arguments for the validity of the monotonicity properties (3.42) and (3.43) were obtained from S.C. Borst [private communication], who studied similar monotonicity properties for polling systems with limited service at all queues.

3.5 Influence of system parameters on the optimal schedule

In the previous section properties of optimal Bernoulli schedules have been presented. The light- and heavy-traffic limits discussed in sections 3.4.1 and 3.4.2 contribute to the insight into the optimal Bernoulli schedules. However, they are only valid in the respective limiting cases and are not very useful for determining an optimal Bernoulli schedule for a particular model instance. The partial solution discussed in section 3.4.3 has more practical importance. The optimal value of some of the components of the optimal schedule is given explicitly, leading to a reduction of the dimension of the remaining optimization problem to find the values of the remaining components of $\boldsymbol{q}^*$. However, this problem is solvable only by means of some numerical procedure. Therefore, we have combined the PSA (with the extension to the computation of derivatives)

with some procedure for non-liear optimization to study the influence of system parameters on the optimal Bernoulli schedule.
In the numerical examples considered here, the number of terms of the power series that has been computed depends on the offered load of the system, and varies from $M = 15$ (for $\rho \leq 0.5$) to $M = 40$ (for $\rho \geq 0.8$). In all cases, the estimated accuracy is in the order of magnitude of 0.001. The optimization procedure is based on grid size 0.001, so that all the digits of the presented optimal schedules are significant.
We will now discuss the influence of system parameters on optimal Bernoulli schedules.

Offered load
Optimal Bernoulli schedules have been computed for various values of the offered load; here, the load is varied in such a way that the ratios between the arrival rates remain fixed. Consider the model with the following set of parameters: $s = 3$; $\boldsymbol{\beta}^{(1)} = (0.50, 1.00, 1.50)$; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/3, \sigma_1/3, \sigma_1/3)$; all service times and switch-over times are exponentially distributed; $\boldsymbol{a} = (1/3, 1/3, 1/3)$; $\boldsymbol{c} = (10/55, 15/55, 30/55)$. For this model, $q_1^* = q_3^* = 1$, in agreement with (3.42). Figure 3.1 shows the values of q_2^* as function of ρ for varying values of σ_1. Figure 3.1 confirms the validity of the light-traffic asymptote (3.33) and of the heavy-traffic asymptote (3.41). Note that in the case $\sigma_1 = 0$ we have $q_2^* = 0$, in agreement with (3.43).

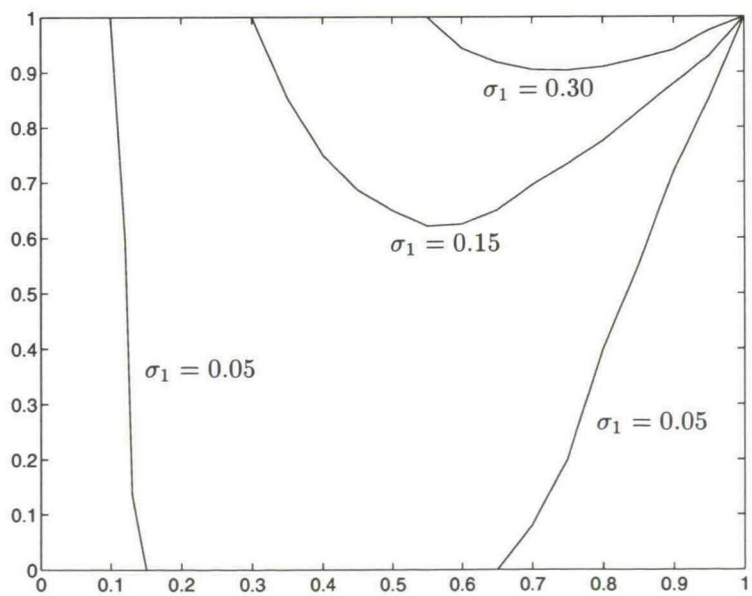


Figure 3.1: Optimal value of q_2^* as function of the offered load; $\sigma_1 > 0$.

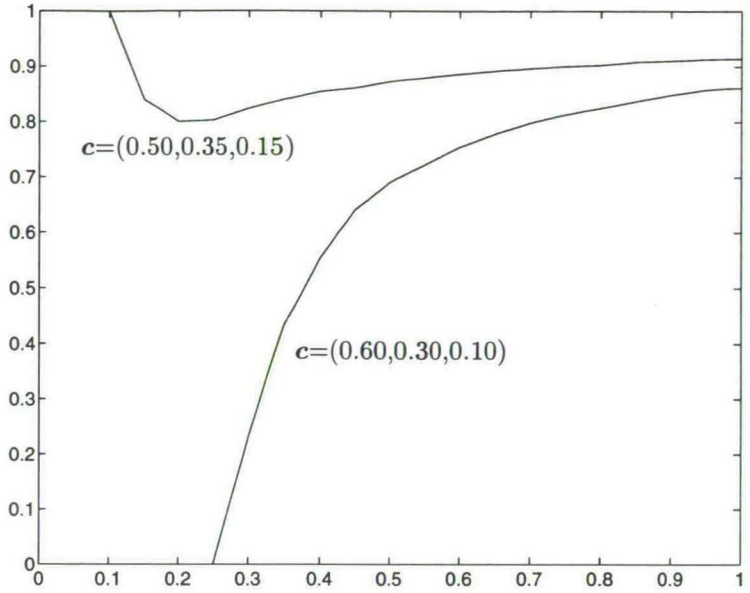


Figure 3.2: Optimal value of q_2^* as function of the offered load; $\sigma_1 = 0$.

As discussed in section 3.3, the characteristics of the optimal schedule as function of ρ in the case $\sigma_1 = 0$ may differ from those in the case $\sigma_1 > 0$. To illustrate this, optimal schedules for varying values of the offered load have been computed for the model with the following set of parameters: $s = 3$; $\boldsymbol{\beta}^{(1)} = (0.50, 1.00, 1.50)$; all service times at Q_1 and Q_3 are exponentially distributed and the service times at Q_2 are 2-phase Coxian distributed with squared coefficient of variation 4; $\boldsymbol{\sigma}^{(1)} = (0.00, 0.00, 0.00)$; $\boldsymbol{a} = (1/3, 1/3, 1/3)$. Two different cost functions are considered, being $\boldsymbol{c} = (0.60, 0.30, 0.10)$ and $\boldsymbol{c} = (0.50, 0.35, 0.15)$. In agreement with (3.42) and (3.43), in both cases we have $q_1^* = 1$ and $q_3^* = 0$. Figure 3.2 shows q_2^* as function of the offered load for both cost functions. Figure 3.2 confirms the validity of the light-traffic asymptotes (3.31), (3.32). Figure 3.2 also illustrates the fact that in the case $\sigma_1 = 0$ the heavy-traffic asymptote (3.41) does not necessarily hold, and that some of the components of the heavy-traffic asymptote of $\boldsymbol{q}^*$ may have values in the interior of the interval [0,1], as remarked at the end of section 3.4.2.

We shall now discuss the influence of the service-time distributions and the switch-over time distributions. In general, the mean waiting times depend on higher moments of the service-time and switch-over time distributions. However, the mean waiting times depend on the service-time and switch-over time distributions primarily through the first two moments of the service times and the switch-over times, as illustrated in Table 1 of [22] and section 5 of [20], re-

spectively. Numerical experiments of Borst et al. [35] with polling models with fixed service limits also support these observations. Therefore, we only consider the influence of the first two moments of the service and switch-over times on the optimal schedule. The observations are based on numerous numerical experiments and have been confirmed in all cases considered. We emphasize that the observations provide only rough guidelines to simplify the problem (e.g. to reduce the size of the state space) and we do not pretend that these observations remain valid for all values of the model parameters.

Service-time distributions
Numerical experience has indicated that the optimal schedules and in particular, the optimal cost, seem to depend primarily on the second moments of the service-time distributions through β_2, rather than on the second moments of the individual service-time distributions, $\boldsymbol{\beta}^{(2)}$. This observation is supported by the PCL (3.26) and by numerical experiments done by Borst et al. [35].
As an illustration, the value of $\boldsymbol{q}^*$ has been computed for combinations of the individual second moments of the service-time distributions, $\boldsymbol{\beta}^{(2)}$, where the second moment of the service time distribution of an arbitrary customer, β_2, is kept fixed. Table 3.1 shows the value of $\boldsymbol{q}^*$ for different combinations $\boldsymbol{\beta}^{(2)}$, which are constructed in such a way that $\beta_2 = 10.00$ in all considered cases. The other parameters are taken to be: $s = 3$; $\boldsymbol{\beta}^{(1)} = (1.00, 2.00, 3.00)$; all service times are 2-phase Coxian distributed; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/3, \sigma_1/3, \sigma_1/3)$; all switch-over times are exponentially distributed; $\boldsymbol{a} = (1/6, 1/6, 1/6)$; $\rho = 0.8$; $\boldsymbol{c} = (0.40, 0.25, 0.35)$. In agreement with (3.42) we have $q_1^* = 1.00$. Table 3.1 shows the results for $\sigma_1 = 0.03$, 0.15 and 1.50, respectively. The optimal schedule is equal to (1.00,1.00,1.00) for σ_1 large enough, and hence, the observation remains valid for large values of σ_1.

	$\sigma_1 = 0.03$		$\sigma_1 = 0.15$		$\sigma_1 = 1.50$	
$\boldsymbol{\beta}^{(2)}$	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$
(2.0, 8.0,20.0)	(0.26,0.00)	8.22	(0.59,0.22)	9.06	(1.00,0.91)	13.48
(2.0,10.0,18.0)	(0.25,0.00)	8.23	(0.57,0.21)	9.06	(1.00,0.90)	13.52
(4.0, 8.0,18.0)	(0.27,0.00)	8.22	(0.60,0.22)	9.06	(1.00,0.91)	13.46
(1.5, 6.0,22.5)	(0.28,0.00)	8.21	(0.62,0.23)	9.05	(1.00,0.90)	13.44
(1.5,15.0,13.5)	(0.25,0.00)	8.24	(0.54,0.21)	9.09	(1.00,0.90)	13.61

Table 3.1: Optimal schedule for different values of $\boldsymbol{\beta}^{(2)}$, with β_2 fixed.

The influence of the second moment of the service-time distribution of an arbitrary customer, β_2, on the optimal schedule seems to be rather unpredictable. In some cases components of the optimal schedule decrease for increasing values of β_2, whereas in other cases components increase for increasing β_2. Moreover, the optimal cost increases when β_2 is increased.
To illustrate this, the optimal schedules have been computed for different val-

ues of β_2 for two different models. The first model is determined by the following set of parameters: $s = 3$; $\boldsymbol{\beta}^{(1)} = (1.00, 2.00, 3.00)$; all service times at queues 1 and 2 are exponentially distributed, and the service times at queue 3 are 2-phase Coxian distributed with squared coefficient of variation α; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/3, \sigma_1/3, \sigma_1/3)$; all switch-over times are exponentially distributed; $\boldsymbol{a} = (1/6, 1/6, 1/6)$; $\rho = 0.8$; $\boldsymbol{c} = (0.40, 0.25, 0.35)$. In agreement with (3.42) we have $q_1^* = 1.00$. Table 3.2 shows the optimal schedules for $\alpha = 0.25$, 0.50, 1.00, 2.00 and 4.00, and for $\sigma_1 = 0.03$ and 0.15, respectively.

	$\sigma_1 = 0.03$		$\sigma_1 = 0.15$	
α	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$
0.25	(0.09,0.00)	5.90	(0.52,0.15)	6.62
0.50	(0.15,0.00)	6.50	(0.54,0.17)	7.25
1.00	(0.23,0.00)	7.69	(0.58,0.20)	8.51
2.00	(0.33,0.00)	10.07	(0.64,0.27)	11.01
4.00	(0.43,0.00)	14.81	(0.74,0.31)	15.95

Table 3.2: Influence of variability of the service times on the optimal schedule.

For the second model the parameters are: $s = 3$; $\boldsymbol{\beta}^{(1)} = (0.50, 1.00, 1.50)$; all service times at Q_1 and Q_3 are exponentially distributed, and the service times at Q_2 are 2-phase Coxian distributed with squared coefficient of variation α; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/3, \sigma_1/3, \sigma_1/3)$; all switch-over times are exponentially distributed; $\boldsymbol{a} = (1/3, 1/3, 1/3)$; $\rho = 0.8$; $\boldsymbol{c} = (0.60, 0.30, 0.10)$. In agreement with (3.42) we have $q_1^* = 1.00$. Table 3.3 shows the optimal schedules for $\alpha = 1.00$, 2.00, 3.00 and 4.00, and $\sigma_1 = 0.03$ and 0.15, respectively.

	$\sigma_1 = 0.03$		$\sigma_1 = 0.15$	
α	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$
1.00	(0.99,0.00)	2.45	(1.00,0.00)	2.81
2.00	(0.91,0.00)	2.80	(0.95,0.00)	3.18
3.00	(0.86,0.00)	3.13	(0.91,0.00)	3.54
4.00	(0.83,0.00)	3.45	(0.87,0.00)	3.89

Table 3.3: Influence of variability of the service times on the optimal schedule.

Switch-over time distributions

In general, the mean waiting times may depend strongly on the switch-over time distributions. However, as discussed in section 5 of [21], the mean waiting times depend on the switch-over time distributions mainly through the first two moments of the *total* switch-over times during one cycle of the server along the queues. Consequently, as for the first moments of the switch-over times,

the optimal Bernoulli schedule depends on $\boldsymbol{\sigma}^{(1)}$ primarily through the first moment σ_1 of the total switch-over time per cycle. Moreover, increasing the mean switch-over time per cycle generally leads to an increase of the components of $\boldsymbol{q}^*$.

To illustrate this, Table 3.4 shows the optimal Bernoulli schedules for varying combinations $\boldsymbol{\sigma}^{(1)}$ of the individual mean switch-over times; the total switch-over times consist of three i.i.d. exponential phases, each with mean $\sigma_1/3$ and hence, are Erlangian-3 distributed with mean σ_1 in all considered cases. The other model parameters are: $s = 3$; $\boldsymbol{\beta}^{(1)} = (1.00, 2.00, 3.00)$; all service times are exponentially distributed; $\boldsymbol{a} = (1/6, 1/6, 1/6)$; $\rho = 0.8$; $\boldsymbol{c} = (0.40, 0.25, 0.35)$. In agreement with (3.42) we have $q_1^* = 1.00$. Table 3.4 shows the results for $\sigma_1 = 0.15$, 0.45 and 1.50, respectively.

	$\sigma_1 = 0.15$		$\sigma_1 = 0.45$		$\sigma_1 = 1.50$	
$\boldsymbol{\sigma}^{(1)}$	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$
(r, r, r)	(0.58,0.21)	8.51	(0.91,0.66)	9.87	(1.00,0.91)	12.83
($3r$, 0, 0)	(0.58,0.20)	8.51	(0.91,0.66)	9.87	(1.00,0.91)	12.82
(0,$3r$, 0)	(0.58,0.20)	8.51	(0.91,0.66)	9.87	(1.00,0.91)	12.85
(0, 0,$3r$)	(0.58,0.20)	8.51	(0.91,0.66)	9.87	(1.00,0.91)	12.83

Table 3.4: Optimal schedule for different values of $\boldsymbol{\sigma}^{(1)}$, with σ_1 and σ_2 fixed.

As far as the second moments of the switch-over time distributions are concerned, we have found out that their effect on the optimal schedule is generally negligible and moreover, that the optimal cost increases when σ_2 is increased. To illustrate this, consider the same model as in Table 3.4 with mean switch-over times $\boldsymbol{\sigma}^{(1)}=(\sigma_1, 0.00, 0.00)$. The optimal schedules have been computed for varying values of $\alpha := (\sigma_2 - {\sigma_1}^2)/{\sigma_1}^2$, which is equal to the squared coefficients of variation of the switch-over times, minus 1. Table 3.5 shows the results for $\sigma_1 = 0.15$, 0.50, 1.50.

	$\sigma_1 = 0.15$		$\sigma_1 = 0.50$		$\sigma_1 = 1.50$	
α	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$	(q_2^*, q_3^*)	$C(\cdot)$
0.25	(0.58,0.20)	8.50	(0.92,0.69)	10.02	(1.00,0.91)	12.76
2.00	(0.58,0.21)	8.61	(0.92,0.69)	10.42	(1.00,0.91)	14.03
4.00	(0.58,0.21)	8.74	(0.93,0.69)	10.88	(1.00,0.90)	15.46
10.00	(0.58,0.23)	9.12	(0.93,0.69)	12.22	(1.00,0.90)	19.67
20.00	(0.60,0.23)	9.76	(0.94,0.70)	14.43	(1.00,0.91)	26.11

Table 3.5: Optimal schedule for different values of σ_2, with σ_1 fixed.

As noted before (below Table 3.1), in the cases considered in Tables 3.4 and 3.5, the optimal schedule is equal to (1.00,1.00,1.00) for σ_1 large enough, so

that the observations remain valid for large values of r.
The foregoing suggests that replacing constant switch-over times by exponentially distributed switch-over times may yield good approximations for the optimal schedules for models with constant switch-over times.

3.6 Approximation

At the beginning of section 3.5 we have discussed a numerical approach to achieve an accurate approximation for the optimal Bernoulli schedule, based on the use of the PSA. The main disadvantage of this approach is the fact that the time and memory requirements increase exponentially with increasing number of queues and hence, its use is restricted to rather small systems. In this section we will propose a simple and fast approach to approximate the optimal Bernoulli schedule, that requires negligible computation time and memory space and is therefore applicable to fairly large systems. This approximate optimal Bernoulli schedule may serve as a starting point for a more accurate optimization procedure based on the use of the PSA.

The approach is based on a simple mean waiting-time approximation (instead of on the use of the PSA). Combined with some classical non-linear optimization procedure (such as the conjugate gradient methods) it yields an approximation for the optimal Bernoulli schedule.
Consider the following mean waiting-time approximation (cf. section 6.7 of [163]):

$$EW_i \approx \frac{(1-\rho+\rho_i)-q_i\rho_i(2-\rho)}{1-\rho[1+\sigma_1 a_i(1-q_i)]}x; \tag{3.48}$$

the quantity x is determined by substituting (3.48), for $i = 1, \ldots, s$, into the PCL (3.26), leading to the following approximation for the mean waiting times at the queues: for $i = 1, \ldots, s$,

$$\begin{aligned} EW_i \approx & \frac{(1-\rho+\rho_i)-q_i\rho_i(2-\rho)}{\sum_{j=1}^{s}\rho_j[(1-\rho+\rho_i)-q_i\rho_i(2-\rho)]} \\ & \times \frac{\frac{\rho^2}{1-\rho}\frac{\beta_2}{2\beta_1}+\rho\frac{\sigma_2}{2\sigma_1}+\frac{\sigma_1}{1-\rho}\sum_{i=1}^{s}\rho_i^2(1-q_i)+\frac{\sigma_1}{2(1-\rho)}\left[\rho^2-\sum_{i=1}^{s}\rho_i^2\right]}{1-\rho[1+\sigma_1 a_i(1-q_i)]}. \end{aligned} \tag{3.49}$$

The approximation is a trivial extension of the PCL-based mean waiting-time approximation proposed in [93] for mixtures of 1-limited and exhaustive (and gated) service disciplines. The latter approach relies on the observation of Everitt [80], who states that the cycle time of Q_i, i.e. the length of the time interval between two successive arrivals of the server to Q_i, has approximately the same second moment for all $i = 1, \ldots, s$.
Although the approximation (3.49) is generally not very accurate for evaluation purposes (cf. numerical results in section 6.7 of [163]), in combination with a non-linear optimization procedure it turns out to give satisfying results for

optimization purposes. The phenomenon that relatively rough approximations lead to quite accurate approximations for optimization in polling models has been observed in other polling studies (cf. e.g. [44], [35]). Apparently, these simple approximations do capture the major factors important for efficient operation.

The mean waiting-time approximation (3.49) depends on the individual service-time distributions only through β_1 and β_2, and similarly, depends on the switch-over time distributions only through σ_1 and σ_2 (cf. (3.26)). As discussed in section 3.3, the optimal schedules and, in particular, the cost of the optimal schedule, are fairly insensitive to the individual service-time and switch-over time distributions.
It is tedious, but straightforward, to verify that the mean waiting time approximation (3.49) satisfies the monotonocity properties in (3.45) and (3.46). Moreover, one may verify that the approximation (3.49) satisfies property (3.44). Hence, following similar arguments as discussed in section 3.4.3, it follows that the Bernoulli schedule $\boldsymbol{q}^*(app)$ that minimizes the cost function $C(\boldsymbol{q})$, as approximated on the basis of the mean waiting-time approximations (3.49), also satisfies properties (3.42) and (3.43).
As noted in section 3.3, in the case $\sigma_1 = 0$ the heavy-traffic asymptote of $\boldsymbol{q}^*$ is generally unknown. Therefore, the heavy-traffic asymptote of the approximated optimum based on (3.49) may differ from the exact heavy-traffic asymptote. Hence, the accuracy of the approximated optimum and, in particular, of the cost belonging to the approximated optimum, may become poor when $\sigma_1 = 0$ and the offered load approaches 1. Moreover, in spite of the fact that for $\sigma_1 > 0$ the heavy-traffic limits of the approximated optima and the actual optima are both equal to 1, the accuracy of the approximated optimum and of the cost belonging to this optimum may become poor, particularly in cases in which $\sigma_1 \approx 0$ and $\rho \approx 1$. This is due to the fact that for $\sigma_1 \approx 0$ the lower bound (cf. (3.40)) of the values of the components of $\boldsymbol{q}$ tends to 1 very slowly, so that for values of ρ close to 1 the lower bound does not force the values of $\boldsymbol{q}$ into a narrow interval.

Figure 3.3 illustrates the behavior of the approximated optimum, $\boldsymbol{q}^*(app)$, as function of the offered load ρ, compared with the optimum $\boldsymbol{q}^*$, obtained by means of the numerical optimization technique discussed in section 3.4. The parameters are: $s = 3$; $\boldsymbol{\beta}^{(1)} = (0.50, 1.00, 1.50)$; $\boldsymbol{\sigma}^{(1)} = (0.05, 0.05, 0.05)$; all service times and switch-over times are exponentially distributed; $\boldsymbol{a} = (1/3, 1/3, 1/3)$; $\boldsymbol{c} = (0.40, 0.25, 0.35)$. In agreement with (3.42) we have $q_1^* = 1.00$.

The quality of the approximation can best be measured by the relative difference between the exact *cost* belonging to $\boldsymbol{q}^*(app)$, $C(\boldsymbol{q}^*(app))$, and the *cost* of the optimum, $C(\boldsymbol{q}^*)$, rather than by the relative accuracy of $\boldsymbol{q}^*(app)$, compared with $\boldsymbol{q}^*$. Table 3.6 gives for the same set of parameters the relative error in the cost function, $err\%$, defined by

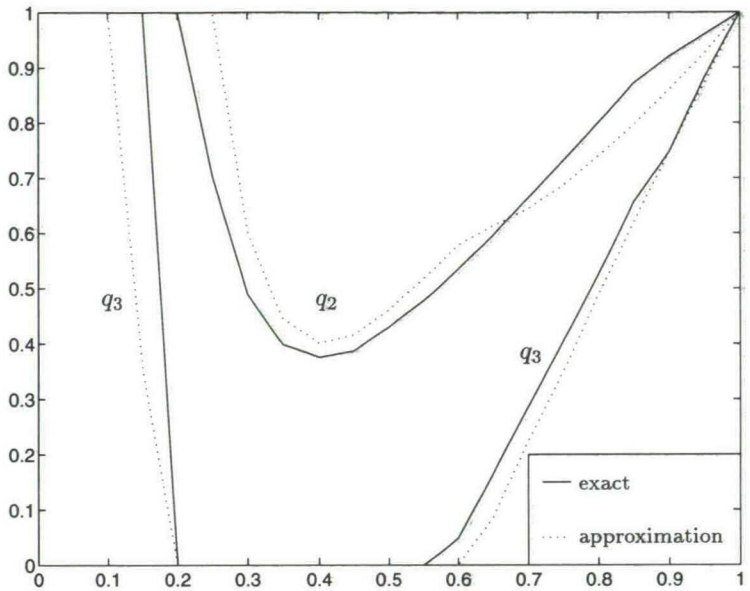


Figure 3.3: Approximated and exact optimum as function of the offered load.

$$\frac{C(\boldsymbol{q}^*(app)) - C(\boldsymbol{q}^*)}{C(\boldsymbol{q}^*)} \times 100. \tag{3.50}$$

From Figure 3.1 one may observe that the approximated schedule may differ considerably from the optimum when the load is rather small. However, as is also illustrated in Table 3.6, the relative error in the cost associated with these optima is small. This is due to the fact that for lightly-loaded systems the cost function (3.2) is rather flat as function of $\boldsymbol{q}$.

To check the quality of the approximation for larger systems, consider a 6-queue model defined by the following set of system parameters: $s = 6$; $\boldsymbol{\beta}^{(1)} = (0.60, 0.80, 1.00, 1.20, 1.40, 1.60)$; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/6, \sigma_1/6, \sigma_1/6, \sigma_1/6, \sigma_1/6, \sigma_1/6)$; all service times and switch-over time are exponentially distributed; $a_i = 10/66$, $i = 1, \ldots, 6$; $\boldsymbol{c} = (0.65, 0.07, 0.07, 0.07, 0.07, 0.07)$. In agreement with (3.42) we have $q_1^* = 1.00$. Tables 3.7 and 3.8 show the other components of the approximated optimum, $\boldsymbol{q}_{2-6}^*(app)$ and of the exact optimum $\boldsymbol{q}^*$ for varying values of the offered load ρ, for $\sigma_1 = 0.30$ and 0.60, respectively.
The relative error in the cost function, which has been computed according to (3.50), is typically less than 0.4% in all cases considered in Tables 3.7 and 3.8. As illustrated in Tables 3.7 and 3.8, for larger systems a main part of the components of the optimal schedule may be equal to 0.00 or 1.00. This may be explained by the fact that for larger systems, the offered load to many of the queues is small, so that the waiting times are fairly independent of the values q_i

ρ	$C(\boldsymbol{q}^*(app))$	$C(\boldsymbol{q}^*)$	$err\%$
0.10	0.236	0.236	0.0
0.15	0.317	0.317	0.0
0.50	1.272	1.272	0.0
0.60	1.836	1.835	0.0
0.70	2.772	2.771	0.0
0.80	4.638	4.635	0.1
0.90	10.231	10.205	0.3
0.95	21.468	21.259	1.0

Table 3.6: Cost of approximated and exact optima.

ρ	$\boldsymbol{q}^*_{2-6}(app)$	$C(\cdot)$	$\boldsymbol{q}^*_{2-6}$	$C(\cdot)$
0.50	(0.23,0.00,0.00,0.00,0.00)	1.34	(0.71,0.00,0.00,0.00,0.00)	1.34
0.60	(0.55,0.00,0.00,0.00,0.00)	1.83	(0.85,0.00,0.00,0.00,0.00)	1.83
0.70	(0.84,0.00,0.00,0.00,0.00)	2.58	(1.00,0.22,0.00,0.00,0.00)	2.57
0.80	(1.00,0.29,0.00,0.00,0.00)	3.98	(1.00,0.43,0.00,0.00,0.00)	3.97

Table 3.7: Approximated and exact optima; $\sigma_1 = 0.30$.

for those queues. Consequently, the cost function (3.2) is rather flat as function of q_i and hence, may be either increasing or decreasing in q_i over the interval [0,1], so that the optimal value of q_i is equal to either 0.00 or 1.00.

In general, the optimal schedule depends on the order in which the queues are visited, whereas the approximated optimum based on (3.49) does *not* depend on the order in which the queues are placed. However, the differences in the optimal schedules and in particular, in the cost associated with the optimal schedules, have turned out to be small.
To illustrate this, consider the model with the following parameters: $s = 6$; $\boldsymbol{\beta}^{(1)} = (0.50, 1.50, 1.50, 1.50, 1.50, 1.50)$; $\boldsymbol{\sigma}^{(1)} = (0.10, 0.10, 0.10, 0.10, 0.10, 0.10)$; all service times and switch-over times are exponentially distributed; $\boldsymbol{a} =$

ρ	$\boldsymbol{q}^*_{2-6}(app)$	$C(\cdot)$	$\boldsymbol{q}^*_{2-6}$	$C(\cdot)$
0.50	(1.00,0.46,0.00,0.00,0.00)	1.68	(1.00,0.36,0.00,0.00,0.00)	1.68
0.60	(1.00,0.56,0.00,0.00,0.00)	2.26	(1.00,0.56,0.00,0.00,0.00)	2.26
0.70	(1.00,0.69,0.19,0.00,0.00)	3.19	(1.00,0.71,0.23,0.00,0.00)	3.19
0.80	(1.00,0.78,0.50,0.28,0.11)	5.03	(1.00,0.78,0.50,0.28,0.13)	5.03

Table 3.8: Approximated and exact optima; $\sigma_1 = 0.60$.

$(1/8, 1/8, 1/8, 1/8, 1/8, 1/8)$; $\boldsymbol{c} = (3/8, 1/8, 1/8, 1/8, 1/8, 1/8)$. In agreement with (3.42) we have $q_1^* = 1.00$. Table 3.9 shows the approximated optimum $\boldsymbol{q}^*(app)$ and the exact optimum $\boldsymbol{q}^*$ for varying values of the offered load ρ; the relative error in the cost function ($err\%$) is computed according to (3.50). Note that $q_i^*(app) = q_2^*(app)$, $i = 2, \ldots, 6$, for all values of ρ; hence, the approximated optimum $\boldsymbol{q}^*(app)$ is indicated by the scalar $q_{2-6}^*(app)$.

ρ	$q_{2-6}^*(app)$	$C(\cdot)$	$\boldsymbol{q}_{2-6}^*$	$C(\cdot)$	$err\%$
0.30	1.00	1.09	(1.00,1.00,1.00,1.00,1.00)	1.09	0.0
0.40	1.00	1.50	(0.79,0.77,0.76,0.76,0.76)	1.50	0.0
0.50	0.95	2.08	(0.66,0.62,0.59,0.57,0.56)	2.06	0.1
0.60	0.82	2.92	(0.66,0.60,0.57,0.55,0.53)	2.90	0.7
0.70	0.80	4.30	(0.72,0.67,0.64,0.62,0.60)	4.28	0.5
0.80	0.84	7.04	(0.86,0.78,0.73,0.72,0.70)	7.01	0.4

Table 3.9: Approximated and exact optima.

3.7 Concluding remarks

In this chapter we have applied the PSA to determine optimal Bernoulli schedules. Because the time and memory requirements needed for this procedure are considerable, we proposed a simple and fast-to-evaluate approximation for optimal Bernoulli schedules, and used the PSA to check the accuracy of these approximated optima.

An alternative approach which is much more generally applicable, also for models for which no simple approximation such as (3.49) is available, is to use the PSA with a small number of terms to find the neighborhood of the optimal schedule with reduced computational effort, and then proceed with the PSA with more terms to locally improve the optimal schedule.

	$\rho = 0.7$		$\rho = 0.8$	
M	$\boldsymbol{q}^*(M)$	$C(\cdot)$	$\boldsymbol{q}^*(M)$	$C(\cdot)$
10	(1.00,0.67,1.00.0.27)	1.978	(0.50,0.51,0.50,0.50)	3.455
20	(1.00,0.65,1.00.0.27)	1.978	(1.00,0.90,1.00,0.57)	3.214
30	(1.00,0.65,1.00.0.26)	1.978	(1.00,0.83,1.00,0.54)	3.210
40	(1.00,0.65,1.00.0.26)	1.978	(1.00,0.82,1.00,0.54)	3.210
80	(1.00,0.65,1.00.0.26)	1.978	(1.00,0.82,1.00,0.54)	3.210

Table 3.10: Optimal Bernoulli schedules for different values of M.

As an illustration of this procedure, consider the polling model in which the

server visits the queues in periodic order 1,2,1,3,1,2,1,3 etcetera, $s = 3$, $\boldsymbol{a} = (1/6, 1/3, 1/2)$, $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$, $\boldsymbol{\sigma}^{(1)} = (0.05, 0.05, 0.05, 0.05)$, $\boldsymbol{c} = (0.60, 0.20, 0.20)$, and in which the service times and the switch-over times are exponentially distributed. Table 3.10 shows the computed values of the optimal Bernoulli schedule $\boldsymbol{q}^*(M)$ as function of the number of terms of the power series (M) which are computed for $\rho = 0.7$ and $\rho = 0.8$. The initial Bernoulli schedule is taken to be (0.50,0.50,0.50,0.50). The cost $C(\cdot)$ belonging to these computed optima have been evaluated with the PSA using $M = 100$ terms.
To illustrate the above-mentioned alternative procedure, let us consider the case $\rho = 0.8$. Table 3.10 suggests that one could use the PSA with $M = 20$ to find a solution in the neighborhood of the optimum, $\boldsymbol{q}^*(20) = (1.00, 0.90, 1.00, 0.57)$, rather quickly and then use the PSA with $M = 30$ or $M = 40$ to locally improve this solution to find an optimum.
We reemphasize that this procedure is generally applicable for optimization by means of the PSA, and goes far beyond the optimization problem considered in this chapter.

Chapter 4

Polling systems with Markovian server routing

4.1 Introduction

In polling models it is typically assumed that the server visits the queues in cyclic order. However, in many cases it is more natural to visit particular queues more frequently than others, e.g. when the queues are not equally loaded. Accordingly, a number of generalizations of the cyclic visit order has been considered in the literature. One generalization is *periodic* polling, in which the server visits the queues periodically according to a fixed service order table, commonly referred to as a polling table. In this way, queues can be given higher priority by listing them more often on the polling table. Alternatively, the server may be routed along the queues according to a *probabilistic* mechanism. An example of such a random mechanism is the so-called Markovian server routing, in which the visit order may be determined by some discrete-time Markov chain. The routing probabilities may be used to give, implicitly, relative priorities to the queues. In this perspective, Markovian polling can be viewed as the *stochastic counterpart* of periodic polling.
In this chapter we study the performance of polling models with Markovian server routing. We compare the performance of Markovian polling models with the system performance under periodic polling. In some applications the routing probabilities may be used as decision variables to assign relative priorities to the queues. This opens possibilities for optimization of the system performance with respect to the routing probabilities. We address the problem of finding optimal combinations of routing probabilities, and investigate characteristics of optimal routing matrices.

Polling models with Markovian server routing find a number of specific applications. For instance, they may be used to model distributed systems, such as a shared broadcast channel where from time to time a decision has to be

made as to who gets the right for transmission. These decisions are usually based on some probabilistic algorithms, rather than on a fixed order (cf. [104]). Alternatively, polling models with Markovian server routing may also be used to predict the expected delay in an exhaustive slotted ALOHA system. In such a system, a station is granted the exclusive right to transmit during some time period. When a transmitting station no longer reserves the channel, some or all stations start contending to seize the channel. Both the length of the contention period and the next station that will seize the channel are random (cf. [116]). Markovian polling is also useful for the modeling of the so-called Orwell slotted-ring protocol. In this protocol, a number of unit-buffer slots of equal length rotate around a ring, and a packet in a slot filled by a station is addressed to some other station with a certain probability, where it is emptied and passed on empty to the next downstream station (cf. [132], [165]). This is a major difference from other slotted-ring protocols, where a slot can be released only by the station that filled it. As another alternative, polling models with Markovian server routing can be used to model material handling systems such as an Automated Guided Vehicle (AGV) system in which a single vehicle serves a manufacturing cell by moving loads from one machining center to another. When the AGV delivers a load to the center, it inspects the output buffer of that center to determine if there are any loads waiting to be transported. If so, the AGV takes some amount of time to pick up load from this output buffer, and a certain amount of time to transport the load and deliver it at its destination, and the AGV polls the output buffer of the center which receives the load. Otherwise, the AGV switches to poll the next center in some order (cf. [49]).

In the literature, only a few papers have been devoted to the analysis of polling models with probabilistic server routing. Kleinrock and Levy [104] analyze the behavior of so-called random polling models, in which after a departure from an arbitrary queue the server is routed to the queue j with some given probability p_j, irrespective of the queue it has just departed from. For infinite-buffer models in which either all queues are served according to the gated service discipline or in which all queues are served exhaustively, Kleinrock and Levy give the mean waiting times at the queues as the solution of a system of linear equations. For symmetrical models with 1-limited service, they determine a closed-form expression for the mean waiting time. Boxma and Weststrate [48] introduce the more general class of polling models with Markovian server routing in which, after a departure of the server from queue i, the server is routed towards queue j with probability $p_{i,j}$. For this class of models, with mixtures of exhaustive, gated and 1-limited service, a pseudo-conservation law (PCL) is derived in [48]. It should be noted that the cyclic server routing is contained within the class of Markovian routing mechanisms (with $p_{i,j} = 1$ if $j = i + 1$ and 0 otherwise), whereas the cyclic routing does *not* occur as a special case of random polling. Srinivasan [153] derives a PCL for polling models with Markovian server routing, in which the routing probabilities may depend on whether customers have been served during the last visit of the server to a

queue. Chung et al. [63] analyze Markovian polling models with unit buffers. They derive exact expressions for the generating function of the joint queue length at polling instants, the Laplace-Stieltjes Transforms (LSTs) of the waiting times and the LST of the cycle-time distribution of each queue.

The analysis of polling models with probabilistic server routing appears to be more involved than the analysis of polling models with periodic server routing. Even Markovian polling models in which each of the service disciplines satisfies the Additivity Property can *not* be included into the framework of Multi-Type Branching Processes (MTBPs), giving a full characterization of the joint queue length at embedded epochs (cf. section 1.2.3 for a more detailed discussion). As a consequence, the efficient algorithms that have recently been developed to compute the moments of the waiting times for MTBP-type models (cf. [106], [107], [154]) are not applicable to polling models with Markovian routing.
In spite of the fact that polling models with probabilistic server routing are generally more involved than their periodic counterparts, some numerical algorithms, though less effective, have been developed to determine the moments of the waiting times at the queues. For models in which either all queues are served exhaustively or all queues are served according to the gated service discipline, Weststrate [171] derives a set of $O(s^3)$ linear equations to obtain the mean waiting times at the queues (cf. also [153]). In their paper on Markovian polling with unit-sized buffers, Chung et al. [63] derive a set of $O(2^s)$ linear equations to determine the mean waiting times. However, for polling models with probabilistic server routing that are not covered in these references, to the best of the author's knowledge, no alternative algorithms are available to compute performance measures concerning the queue-length distributions.

In this chapter we show how the applicability of the power-series algorithm (PSA) can be extended to analyze a general class of polling models with Markovian server routing. Throughout, these models will be alternatively referred to as Markov-polling models or polling models with Markovian polling. The extension of the PSA is used for making comparisons between Markov-polling models on the one hand and periodic-polling models on the other hand. The only available results about how Markov-polling models relate to periodic-polling models concern fully symmetrical models in which *all* switch-over times (including those from a queue to itself) are equal. For these models it can be shown that in case of cyclic polling the mean waiting time is smaller than in case of random polling (with all routing probabilities equal). However, in many cases the symmetry assumption is far from realistic, e.g. when particular queues are heavier loaded than others, or when the switch-over times represent physical movement from one station to another while the stations are not equidistant. For asymmetrical models, little is known about the relative performance of the system under periodic and probabilistic server routing.
To make a *comparison* of the performance of polling models under *periodic* and *Markovian* polling we have performed numerous numerical experiments with the PSA. The results indicate that the mean total amount of work in the system

is structurally smaller under periodic polling than under Markovian polling. However, the experiments with the PSA have shown that a similar dominance relation is not generally valid for the individual mean waiting times.
In addition, we consider the problem of characterizing combinations of routing probabilities that minimize the mean amount of waiting work in the system (cf. (1.2)). However, the dimension of the *optimization* problem grows quadratically in the number of queues, making numerical procedures based on standard techniques for non-linear optimization very time consuming when the number of queues becomes large. Therefore, we focus on finding *qualitative*, instead of quantitative, properties of optimal routing matrices. For fully symmetrical models, Liu et al. [125] have shown that each cyclic server routing (which occurs as a special case of Markovian server routing) is optimal. Numerical experiments with the PSA suggest that the cyclic service order is a stable optimum in the sense that it remains optimal for slight perturbations of the system parameters. When the model becomes even more asymmetrical, the cyclic service order may become suboptimal. We observe the tendency of the optimal solution towards (partially) deterministic routing; that is, for relatively many queues i there exists a specific queue k_i such that the optimal routing probabilities are equal to $p_{i,j} = 1$ if $j = k_i$ and 0 otherwise. In addition, numerical experiments suggest that optimal routing matrices can be roughly classified into a limited number of types of solutions, each having specific characteristics that can be interpreted fairly easily. Based on the obtained insights, we give some guidelines for the construction of optimal routing matrices.

The remainder of this chapter is organized as follows. Section 4.2 contains a description of the model. In section 4.3 we show how the present Markov-polling model can be analyzed by means of the PSA, with the computation of derivatives with respect to the routing probabilities. In section 4.4 we apply the PSA to establish a comparison between the performance of Markov-polling models and periodic-polling models. In section 4.5 we investigate properties of optimal combinations of routing probabilities. Section 4.6 contains some concluding remarks and addresses some topics for future research.

4.2 Model description

Consider the basic polling model discussed in section 1.3, with s infinite-buffer queues, $Q_1, \ldots, Q_s$, and Poisson arrival processes with rates $\lambda_i = a_i \rho$. The service times of customers at Q_i are Coxian distributed with parameters $\pi_i^{1,\xi}, \mu_i^{1,\xi}, \Psi_i^1$, $\xi = 1, \ldots, \Psi_i^1$, $i = 1, \ldots, s$. The queues are served according to the Bernoulli schedule $\boldsymbol{q} = (q_1, \ldots, q_s)$. The server visits the queues according to a Markovian polling scheme with routing matrix $\mathbf{P} = (p_{i,j})$; that is, after a departure of the server from Q_i the server starts to move to Q_j with probability $p_{i,j}$, $i, j = 1, \ldots, s$. In this way, the process of successive visits of the server to the various queues can be described as a discrete-time Markov chain $D = \{d_k,\ k = 0, 1, \ldots\}$ with state space $\{1, \ldots, s\}$, where $\{d_k = i\}$ denotes the

event that the k-th visited queue is Q_i, $i = 1, \ldots, s$, $k = 0, 1, \ldots$. Throughout, it is assumed that D is irreducible. The times needed by the server to move from Q_i to Q_j are Coxian distributed with parameters $\pi_{i,j}^{0,\xi}, \mu_{i,j}^{0,\xi}, \Psi_{i,j}^{0,\xi}$, $\xi = 1, \ldots, \Psi_{i,j}^{0}$, $i, j = 1, \ldots, s$. Denote by $\sigma_{i,j}^{(k)}$ the k-th moment of the switch-over times to move from Q_i to Q_j, $i, j = 1, \ldots, s$, $k = 1, 2$. Because D is an irreducible Markov chain on a finite state space, it possesses a stationary distribution $\{\omega_i,\ i = 1, \ldots, s\}$, which is uniquely determined by the following set of equations (cf. e.g. Theorem 4.1 of [148]):

$$\omega_i = \sum_{j=1}^{s} \omega_j p_{j,i} \ (i = 1, \ldots, s); \qquad \sum_{j=1}^{s} \omega_j = 1. \tag{4.1}$$

Necessary and sufficient conditions for the stability of the system have been derived in [86]. For the present model with Markovian server routing these conditions read:

$$\rho \left[1 + \frac{\sigma a_i (1 - q_i)}{\omega_i} \right] < 1, \tag{4.2}$$

where

$$\sigma := \sum_{i=1}^{s} \omega_i \sum_{j=1}^{s} p_{i,j} \sigma_{i,j}^{(1)}, \tag{4.3}$$

i.e. the mean of an arbitrary switch-over time.
In the sequel it is assumed that these conditions are satisfied and that the system is in steady state.

4.3 The power-series algorithm

In this section we show how the present model can be analyzed by means of the PSA. We emphasize that, because of the quasi birth-and-death (QBD) structure of the model, the implementation is a special case of the general approach discussed in chapter 2. However, we believe that it is still worthwhile to discuss the present model, because it addresses some interesting aspects of the PSA, that are typical for probabilistic polling and do not occur in periodic polling. Recall that in the general case for each $(k; \boldsymbol{n})$-combination a set of linear equations has to be solved (as many as the size of the supplementary space). However, the specific structure of the present model with Markov polling is explored to make the PSA more efficient in such a way that for each $(k; \boldsymbol{n})$ only smaller sets of equations (as many as the number of empty queues) have to be solved.
The approach follows the same lines as in section 3.3. We define the state probabilities and give the global balance equations for the present model. Then, the state probabilities and their derivatives with respect to the routing probabilities are expressed as power series in the offered load to the system and finally, we derive a computational scheme to compute the coefficients of these power series.

4.3.1 Balance equations

Let $\{\boldsymbol{N}(t) = (N_1(t), \ldots, N_s(t)), t \geq 0\}$ be the joint queue-length process. Evidently, this process is not a Markov process, e.g. because the departure rate depends on whether the server is switching or serving. To transform the process $\{\boldsymbol{N}(t), t \geq 0\}$ into a Markov process, we introduce a triple $(H(t), G(t), \Xi(t))$ of supplementary variables. Let $\{H(t) = h;\ G(t) = 1;\ \Xi(t) = \xi\}$ denote the event that at time t the server is serving at Q_h and that ξ is the current phase number of this service, $h = 1, \ldots, s,\ \xi = 1, \ldots, \Psi_h^1,\ t \geq 0$. Moreover, let the event $\{H(t) = h;\ G(t) = -g;\ \Xi(t) = \xi\}$ indicate that at time t the server is switching from Q_g towards Q_h, and that ξ is the current phase number of this switch-over, $g, h = 1, \ldots, s,\ \xi = 1, \ldots, \Psi_{g,h}^0,\ t \geq 0$. For ease of the discussion, it is assumed that the supplementary space is the same for all $\boldsymbol{n} \in \mathbb{N}^s$, and is given by

$$\mathcal{S} := \{1, \ldots, s\} \times \{-s, \ldots, -1, 1\} \times \{1, \ldots, K\}, \tag{4.4}$$

where $K := \max_{i,j}\{\Psi_{i,j}^0, \Psi_i^1\}$. One may verify that the combined process $\{(\boldsymbol{N}(t), H(t), G(t), \Xi(t)), t \geq 0\}$ is a QBD process on the state space $\mathbb{N}^s \times \mathcal{S}$. Denote by $(\boldsymbol{N}, H, G, \Xi)$ random variables with as joint distribution the stationary distribution of $(\boldsymbol{N}(t), H(t), G(t), \Xi(t))$.

Define the state probabilities as follows: for $(\boldsymbol{n}, h, -g, \xi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, h, -g, \xi) := \Pr\{(\boldsymbol{N}, H, G, \Xi) = (\boldsymbol{n}, h, -g, \xi)\}. \tag{4.5}$$

Because of the stability of the system the rate into each state is equal to the rate out of that state. The state probabilities satisfy the following balance equations for the states in which the server is switching (from Q_g to Q_h): for $\boldsymbol{n} \in \mathbb{N}^s,\ g, h = 1, \ldots, s,\ \xi = 1, \ldots, \Psi_{g,h}^0$,

$$\begin{aligned}
&\left[\rho \sum_{j=1}^{s} a_j + \mu_{g,h}^{0,\xi}\right] p(\boldsymbol{n}, h, -g, \xi) = \\
&\mu_{g,h}^{0,\xi+1} p(\boldsymbol{n}, h, -g, \xi+1) I\left\{\xi < \Psi_{g,h}^0\right\} \\
&+ \rho \sum_{j=1}^{s} a_j p(\boldsymbol{n} - \boldsymbol{e}_j, h, -g, \xi) I\{n_j > 0\} \\
&+ \pi_{g,h}^{0,\xi} p_{g,h} \sum_{f=1}^{s} \mu_{f,g}^{0,1} p(\boldsymbol{n}, g, -f, 1) I\{n_g = 0\} \\
&+ \mu_g^{1,1} \pi_{g,h}^{1,\xi} p_{g,h} p(\boldsymbol{n} + \boldsymbol{e}_g, g, 0, 1)\left[1 - q_g I\{n_g > 0\}\right].
\end{aligned} \tag{4.6}$$

The first term at the right-hand side indicates a phase transition in a switch-over time from Q_g to Q_h. The second term corresponds to an arrival while the server is switching from Q_g to Q_h. The third term describes that the server finds Q_g empty upon arrival and immediately starts to move to Q_h. Finally, the fourth term indicates that the server departs from Q_g after service completion of a customer at that queue and proceeds to Q_h.

The global balance equations for the states in which the server is serving (at Q_h) read as follows: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^1$, $n_h > 0$,

$$\begin{aligned}\left[\rho \sum_{j=1}^{s} a_j + \mu_h^{1,\xi}\right] p(\boldsymbol{n}, h, 1, \xi) &= \mu_h^{1,\xi+1} p(\boldsymbol{n}, h, 1, \xi+1) I\left\{\xi < \Psi_h^1\right\} \\ &+ \rho \sum_{j=1}^{s} a_j p(\boldsymbol{n} - \boldsymbol{e}_j, h, 1, \xi) I\left\{n_j > 0\right\} + \pi_h^{1,\xi} \sum_{g=1}^{s} \mu_{g,h}^{0,1} p(\boldsymbol{n}, h, -g, 1) \\ &+ q_h \mu_h^{1,1} \pi_h^{1,\xi} p(\boldsymbol{n} + \boldsymbol{e}_h, h, 1, 1).\end{aligned} \tag{4.7}$$

The first term indicates a phase transition in a service of a customer at Q_h. The second term corresponds to an arrival during the service of a customer at Q_h. The third term describes that the server arrives at Q_h and immediately starts to serve a customer at that queue. The fourth term indicates that after a service completion at Q_h the server immediately starts to serve the next customer at that queue.

Because the server can not be serving at an empty queue, we have: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^1$,

$$p(\boldsymbol{n}, h, 1, \xi) = 0 \text{ if } n_h = 0, \tag{4.8}$$

and according to the law of total probability, we have

$$\sum_{\boldsymbol{n} \in \mathbb{N}^s} \sum_{h=1}^{s} \left\{ \sum_{g=1}^{s} \sum_{\xi=1}^{\Psi_{g,h}^0} p(\boldsymbol{n}, h, -g, \xi) + \sum_{\xi=1}^{\Psi_h^1} p(\boldsymbol{n}, h, 1, \xi) \right\} = 1. \tag{4.9}$$

The set of balance equations (4.6), (4.7), together with the law of total probability (4.9), forms a non-recursively solvable infinite set of linear equations between the state probabilities. The conditions for application of the PSA for the present model (as discussed in section 2.3.2) are satisfied, because of the assumption that the discrete-time Markov process D is irreducible on the state space $\{1, \ldots, s\}$.

4.3.2 Computational scheme

In this section we first express the state probabilities (4.5), and their derivatives with respect to the routing probabilities, as power series in the offered load to the system. Then, we derive a complete computational scheme to calculate the coefficients of these power series.

Using the light-traffic property $p(\boldsymbol{n}, h, -g, \xi) = O(\rho^{|\boldsymbol{n}|})$, for $\rho \downarrow 0$ (cf. (2.5)), we express the state probabilities as power series in ρ as follows: for $(\boldsymbol{n}, h, -g, \xi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, h, -g, \xi) = \rho^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \rho^k b_0(k; \boldsymbol{n}, h, -g, \xi). \tag{4.10}$$

We now define the derivatives of the state probabilities with respect to the routing probabilities $p_{i,j}$. For notational convenience, the s^2 derivatives are ordered linearly as follows: for $r = 1, \ldots, s^2$, $(\boldsymbol{n}, h, -g, \xi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p_r(\boldsymbol{n}, h, -g, \xi) := \frac{\partial}{\partial p_{i,j}} p(\boldsymbol{n}, h, -g, \xi), \tag{4.11}$$

where i, j and r are related through

$$r = (i-1)s + j, \quad i, j = 1, \ldots, s. \tag{4.12}$$

The derivatives of the state probabilities can be expressed as power series in ρ as follows: for $r = 1, \ldots, s^2$, $(\boldsymbol{n}, h, -g, \xi) \in \mathbb{N}^s \times \mathcal{S}$,

$$p_r(\boldsymbol{n}, h, -g, \xi) = \rho^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \rho^k b_r(k; \boldsymbol{n}, h, -g, \xi). \tag{4.13}$$

Because the routing probabilities do not depend on the variable ρ, the coefficients $b_r(k; \boldsymbol{n}, h, -g, \xi)$ can be obtained by termwise differentiation of the coefficients $b_0(k; \boldsymbol{n}, h, -g, \xi)$: for $r = 1, \ldots, s^2$, $(k; \boldsymbol{n}, h, -g, \xi) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$b_r(k; \boldsymbol{n}, h, -g, \xi) := \frac{\partial}{\partial p_{i,j}} b_0(k; \boldsymbol{n}, h, -g, \xi), \tag{4.14}$$

where i, j and r are related through (4.12). Substituting the power-series expansions (4.10) into the balance equations (4.6) and (4.7), and equating corresponding powers of ρ leads to the following sets of linear relations between the coefficients of the power series in (4.10) and (4.11): for $r = 0, 1, \ldots, s^2$, $\boldsymbol{n} \in \mathbb{N}^s$, $h, g = 1, \ldots, s$, $\xi = 1, \ldots, \Psi^0_{g,h}$, $k = 0, 1, \ldots$,

$$\begin{aligned}
&\mu^{0,\xi}_{g,h} b_r(k; \boldsymbol{n}, h, -g, \xi) = \mu^{0,\xi+1}_{g,h} b_r(k; \boldsymbol{n}, h, -g, \xi+1) I\left\{\xi < \Psi^0_{g,h}\right\} \\
&+ \sum_{j=1}^{s} a_j b_r(k; \boldsymbol{n} - \boldsymbol{e}_j, h, -g, \xi) I\{n_j > 0\} \\
&- \sum_{j=1}^{s} a_j b_r(k-1; \boldsymbol{n}, h, -g, \xi) I\{k > 0\} \\
&+ \pi^{0,\xi}_{g,h} p_{g,h} \sum_{f=1}^{s} \mu^{0,1}_{f,g} b_r(k; \boldsymbol{n}, g, -f, 1) I\{n_g = 0\} \\
&+ \pi^{0,\xi}_{g,h} \left[\frac{\partial p_{g,h}}{\partial p_{i,j}}\right] \sum_{f=1}^{s} \mu^{0,1}_{f,g} b_0(k; \boldsymbol{n}, g, -f, 1) I\{r > 0\} I\{n_g = 0\} \\
&+ \mu^{1,1}_{g} \pi^{0,\xi}_{g,h} p_{g,h} b_r(k; \boldsymbol{n} + \boldsymbol{e}_g, g, 1, 1) [1 - q_g I\{n_g > 0\}] I\{k > 0\} \\
&+ \mu^{1,1}_{g} \pi^{0,\xi}_{g,h} \left[\frac{\partial p_{g,h}}{\partial p_{i,j}}\right] b_0(k; \boldsymbol{n} + \boldsymbol{e}_g, g, 1, 1) \\
&\qquad \times [1 - q_g I\{n_g > 0\}] I\{k > 0\} I\{r > 0\};
\end{aligned} \tag{4.15}$$

and for the coefficients corresponding to the states in which the server is serving: for $r = 0, 1, \ldots, s^2$, $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi^1_h$, $k = 0, 1, \ldots$,

$$\begin{aligned}
&\mu_h^{1,\xi} b_r(k;\boldsymbol{n},h,1,\xi) = \mu_h^{1,\xi+1} b_r(k;\boldsymbol{n},h,1,\xi+1) I\{\xi < \Psi_h^1\} \\
&+ \sum_{j=1}^{s} a_j b_r(k;\boldsymbol{n}-\boldsymbol{e}_j,h,1,\xi) I\{n_j > 0\} \\
&- \sum_{j=1}^{s} a_j b_r(k-1;\boldsymbol{n},h,1,\xi) I\{k > 0\} \\
&+ \pi_h^{1,\xi} \sum_{g=1}^{s} \mu_{g,h}^{0,1} b_r(k;\boldsymbol{n},h,-g,1) \\
&+ q_h \mu_h^{1,1} \pi_h^{1,\xi} b_r(k-1;\boldsymbol{n}+\boldsymbol{e}_h,h,1,1) I\{k > 0\}.
\end{aligned} \tag{4.16}$$

For convenience, we rewrite the set of equations (4.15) as follows: for $r = 0,1,\ldots,s^2$, $\boldsymbol{n} \in \mathbb{N}^s$, $h,g = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_{g,h}^0$, $k = 0,1,\ldots$,

$$\begin{aligned}
&\mu_{g,h}^{0,\xi} b_r(k;\boldsymbol{n},h,-g,\xi) = \\
&\pi_{g,h}^{0,\xi} p_{g,h} \sum_{f=1}^{s} \mu_{f,g}^{0,1} b_r(k;\boldsymbol{n},g,-f,1) I\{n_g = 0\} \\
&+ y_r(k;\boldsymbol{n},h,-g,\xi),
\end{aligned} \tag{4.17}$$

where

$$\begin{aligned}
&y_r(k;\boldsymbol{n},h,-g,\xi) := \mu_{g,h}^{0,\xi+1} b_r(k;\boldsymbol{n},h,-g,\xi+1) I\left\{\xi < \Psi_{g,h}^0\right\} \\
&+ \sum_{j=1}^{s} a_j b_r(k;\boldsymbol{n}-\boldsymbol{e}_j,h,-g,\xi) I\{n_j > 0\} \\
&- \sum_{j=1}^{s} a_j b_r(k-1;\boldsymbol{n},h,-g,\xi) I\{k > 0\} \\
&+ \pi_{g,h}^{0,\xi} \left[\frac{\partial p_{g,h}}{\partial p_{i,j}}\right] \sum_{f=1}^{s} \mu_{f,g}^{0,1} b_0(k;\boldsymbol{n},g,-f,1) I\{r > 0\} I\{n_g = 0\} \\
&+ \mu_g^{1,1} \pi_{g,h}^{0,\xi} p_{g,h} b_r(k-1;\boldsymbol{n}+\boldsymbol{e}_g,g,1,1) \\
&\qquad \times [1 - q_g I\{n_g > 0\}] I\{k > 0\} \\
&+ \mu_g^{1,1} \pi_{g,h}^{0,\xi} \left[\frac{\partial p_{g,h}}{\partial p_{i,j}}\right] b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_g,g,1,1) \\
&\qquad \times [1 - q_g I\{n_g > 0\}] I\{k > 0\} I\{r > 0\}.
\end{aligned} \tag{4.18}$$

To derive a computation order for the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$, we need to explore the structure of the set of equations (4.17). To this end, we assign to each $\boldsymbol{n} \in \mathbb{N}^s$ the *null-set* corresponding to $\boldsymbol{n}$ as follows: for $\boldsymbol{n} \in \mathbb{N}^s$,

$$\mathcal{N}_{\boldsymbol{n}}^{(0)} := \{1 \le g \le s \mid n_g = 0\}, \tag{4.19}$$

i.e. the set of empty queues when the joint queue-length vector is $\boldsymbol{n}$. In addition, we define: for $(k;\boldsymbol{n}) \in \mathbb{N}^{1+s}$, $g \in \mathcal{N}_{\boldsymbol{n}}^{(0)}$, $r = 0,1,\ldots,s^2$,

$$C_r(k;\boldsymbol{n},g) := \sum_{f=1}^{s} \mu_{f,g}^{0,1} b_r(k;\boldsymbol{n},g,-f,1). \tag{4.20}$$

By summing both sides of the equations (4.17) over $g = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_{g,h}^0$, we obtain the following set of equations: for $r = 0,1,\ldots,s^2$, $(k;\boldsymbol{n}) \in \mathbb{N}^{1+s}$, $h \in \mathcal{N}_{\boldsymbol{n}}^{(0)}$,

$$C_r(k;\boldsymbol{n},h) = \sum_{g\in\mathcal{N}_{\boldsymbol{n}}^{(0)}} C_r(k;\boldsymbol{n},g)p_{g,h} + \bar{y}_r(k;\boldsymbol{n},h), \tag{4.21}$$

where

$$\bar{y}_r(k;\boldsymbol{n},h) := \sum_{g=1}^{s}\sum_{\xi=1}^{\Psi_{g,h}^0} y_r(k;\boldsymbol{n},h,-g,\xi). \tag{4.22}$$

It should be noted that by introducing the quantities $C_r(k;\boldsymbol{n},g)$ in (4.20), for a *given* triple $(r,k;\boldsymbol{n})$, the set of $|\mathcal{S}|$ linear equations for computing the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$, $(h,-g,\xi)\in\mathcal{S}$ has been *reduced* to the set of $|\mathcal{N}_{\boldsymbol{n}}^{(0)}|$ linear equations for the quantities $C_r(k;\boldsymbol{n},g)$, $g\in\mathcal{N}_{\boldsymbol{n}}^{(0)}$ (cf. (4.21)).

Once the quantities $C_r(k;\boldsymbol{n},g)$, with $g\in\mathcal{N}_{\boldsymbol{n}}^{(0)}$, are known, the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$, $(h,-g,\xi)\in\mathcal{S}$, can be obtained from the following relation (cf. also (4.17)): for $r=0,1,\ldots,s^2$, $(k;\boldsymbol{n})\in\mathbb{N}^{1+s}$, $h,g=1,\ldots,s$, $\xi=1,\ldots,\Psi_{g,h}^0$,

$$\mu_{g,h}^{0,\xi} b_r(k;\boldsymbol{n},h,-g,\xi) = \pi_{g,h}^{0,\xi} p_{g,h} C_r(k;\boldsymbol{n},g) + y_r(k;\boldsymbol{n},h,-g,\xi), \tag{4.23}$$

with $C_r(k;\boldsymbol{n},g) := 0$ for $g\notin\mathcal{N}_{\boldsymbol{n}}^{(0)}$.

We are now ready to define an ordering of the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$ such that they can be determined recursively. Let us first define an ordering for the states with $r=0$. To this end, we adopt the ordering $\prec$ over the $(k;\boldsymbol{n})$-combinations introduced in (2.12): for $(k;\boldsymbol{n},h,-g,\xi)\in\mathbb{N}^{1+s}\times\mathcal{S}$,

$$\begin{aligned}&(k;\boldsymbol{n},h,-g,\xi) \prec (\hat{k};\hat{\boldsymbol{n}},\hat{h},-\hat{g},\hat{\xi})\\ &\text{if } \left[k+|\boldsymbol{n}|<\hat{k}+|\hat{\boldsymbol{n}}|\right] \vee \left[k+|\boldsymbol{n}|=\hat{k}+|\hat{\boldsymbol{n}}| \wedge k<\hat{k}\right].\end{aligned} \tag{4.24}$$

For given $(k;\boldsymbol{n})$ *and* h, we define the following ordering over the couples $(-g,\xi)$, $g=1,\ldots,s$, $\xi=1,\ldots,\Psi_{g,h}^0$, $\hat{\xi}=1,\ldots,\Psi_h^1$,

$$(k;\boldsymbol{n},h,-g,\xi) \prec (k;\boldsymbol{n},h,1,\hat{\xi}); \tag{4.25}$$

thus, for given $(k;\boldsymbol{n},h)$, the coefficients corresponding to the states in which the server is serving are of higher order than those corresponding to states in which the server is switching. In addition, for given $(k;\boldsymbol{n},h)$, the states in which the server is switching are (partially) ordered as follows:

$$(k;\boldsymbol{n},h,-g,\xi) \prec (k;\boldsymbol{n},h,-\hat{g},\hat{\xi}) \text{ if } \left[g\notin\mathcal{N}_{\boldsymbol{n}}^{(0)}\right] \wedge \left[\hat{g}\in\mathcal{N}_{\boldsymbol{n}}^{(0)}\right]; \tag{4.26}$$

thus, the coefficients corresponding to states in which the server is switching after a departure from a non-empty queue are of lower order than those for the states in which the server is moving just after a departure from an empty queue. For given $(k;\boldsymbol{n},h)$ *and* $g=1,\ldots,s$, the states $(k;\boldsymbol{n},h,-g,\xi)$ are ordered as follows: for $(k;\boldsymbol{n})\in\mathbb{N}^{1+s}$, $h,g=1,\ldots,s$, $\xi,\tilde{\xi}=1,\ldots,\Psi_{g,h}^0$,

$$(k;\boldsymbol{n},h,-g,\xi) \prec (k;\boldsymbol{n},h,-g,\hat{\xi}) \text{ if } \xi > \hat{\xi}; \tag{4.27}$$

thus, for the states in which the server is moving from a given queue Q_g towards Q_h are ranked in increasing order with respect to the component ξ as $\Psi^0_{g,h}, \Psi^0_{g,h}-1,\ldots,1$. We emphasize that this ordering, defined by (4.24)-(4.27) is only partial, so that not all couples of vectors $(k;\boldsymbol{n},h,-g,\xi) \in \mathbb{N}^{1+s} \times \mathcal{S}$ are mutually ordered.

Thus far, we have only considered the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$ with $r=0$. To derive an ordering for the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$, $r=0,1,\ldots,s^2$, we extend the partial ordering $\prec$, defined in (4.24)-(4.27), to an ordering of the states $(r,k;\boldsymbol{n},h,-g,\xi)$ as follows (cf. also (2.43)): for $(r,k;\boldsymbol{n},h,-g,\xi)$, $(\hat{r},\hat{k},\hat{\boldsymbol{n}};\hat{h},-\hat{g},\hat{\xi}) \in \{0,1,\ldots,s^2\} \times \mathbb{N}^{1+s} \times \mathcal{S}$,

$$\begin{aligned}&(r,k;\boldsymbol{n},h,-g,\xi) \tilde{\prec} (\hat{r},\hat{k};\hat{\boldsymbol{n}},\hat{h},-\hat{g},\hat{\xi})\\ &\text{if } [r=0 \wedge \hat{r}>0] \vee \left[r=\hat{r} \wedge (k;\boldsymbol{n},h,-g,\xi) \prec (\hat{k},\hat{\boldsymbol{n}};\hat{h},-\hat{g},\hat{\xi})\right].\end{aligned} \tag{4.28}$$

One may verify that under the partial ordering $\tilde{\prec}$ all coefficients in (4.18) are of lower order than $b_r(k;\boldsymbol{n},h,-g,\xi)$, and that all terms at the right-hand side of (4.16) are of lower order with respect to $\tilde{\prec}$ than $b_r(k;\boldsymbol{n},h,1,\xi)$.

Hence, it remains to consider the solvability of the set of equations (4.17) for given $(r,k;\boldsymbol{n})$. To this end, note that for given $(r,k;\boldsymbol{n})$ the set of equations (4.17) is uniquely solvable if and only if the set (4.21) is uniquely solvable. To consider the solvability of (4.21), a distinction has to be made between the empty and non-empty states.

For $\boldsymbol{n} \neq \boldsymbol{0}$, the reduced routing matrix $\mathbf{P}=(p_{i,j})$, $i,j \in \mathcal{N}^{(0)}_{\boldsymbol{n}}$ is substochastic (cf. [148]), which guarantees that the set (4.21) indeed possesses a unique solution $C_r(k;\boldsymbol{n},g)$, $g \in \mathcal{N}^{(0)}_{\boldsymbol{n}}$. To consider the solvability of the set of equations (4.21) for the states with $\boldsymbol{n}=\boldsymbol{0}$, one may verify, by summing both sides over $h \in \mathcal{N}^{(0)}_{\boldsymbol{0}} = \{1,\ldots,s\}$, that for given $(r,k;\boldsymbol{n})$ relations (4.21) form a *dependent* set of equations. One may verify that this set of equations is not contradictive because of a necessary balance between the empty states and states with exactly one customer in the system, implying: for $r=0,1,\ldots,s^2$, $k=0,1,\ldots$,

$$\sum_{h=1}^{s} \bar{y}_r(k;\boldsymbol{0},h) = \sum_{g=1}^{s}\sum_{h=1}^{s}\sum_{\xi=1}^{\Psi^0_{g,h}} y_r(k;\boldsymbol{0},h,-g,\xi) = 0. \tag{4.29}$$

An additional equation follows directly from the law of total probability (4.9), which implies: for $r=0,1,\ldots,s^2$, $k=0,1,\ldots$,

$$\sum_{g=1}^{s}\sum_{h=1}^{s}\sum_{\xi=1}^{\Psi^0_{g,h}} b_r(k;\boldsymbol{0},h,-g,\xi) = Y_r(k), \tag{4.30}$$

where for $r=0,1,\ldots,s^2$, $Y_r(0) := I\{r=0\}$ and for $k=1,2,\ldots$,

$$Y_r(k) := -\sum_{0<|\boldsymbol{n}|\leq k}\sum_{h=1}^{s}\left\{\sum_{g=1}^{s}\sum_{\xi=1}^{\Psi_{g,h}^0} b_r(k-|\boldsymbol{n}|;\boldsymbol{n},h,-g,\xi) + \sum_{\xi=1}^{\Psi_h^1} b_r(k-|\boldsymbol{n}|;\boldsymbol{n},h,1,\xi)\right\}. \tag{4.31}$$

Equation (4.30) can be rewritten in terms of the variables $C_r(k;\mathbf{0},g)$ in the following way: for $r=0,1,\ldots,s^2$, $k=0,1,\ldots$,

$$\sum_{g=1}^{s}\left(\sum_{h=1}^{s}\sum_{\xi=1}^{\Psi_{g,h}^0}\frac{p_{g,h}}{\mu_{g,h}^{0,\xi}}\sum_{\psi=\xi}^{\Psi_{g,h}^0}\pi_{g,h}^{0,\psi}\right)C_r(k;\mathbf{0},g) = Y_r(k) - \sum_{g=1}^{s}\sum_{h=1}^{s}\sum_{\xi=1}^{\Psi_{g,h}^0}\frac{1}{\mu_{g,h}^{0,\xi}}\sum_{\psi=\xi}^{\Psi_{g,h}^0} y_r(k;\mathbf{0},h,-g,\psi). \tag{4.32}$$

Now, it suffices to show that the set of equations (4.21), (4.32) or equivalently, the set (4.17), (4.30), is uniquely solvable. To this end, it should be noted that the coefficients at the left-hand side of these sets of equations are independent of r and k, so that it is sufficient to consider the solvability of these sets of equations for $r=0$ and $k=0$. Then the solvability is readily established by observing that the continuous-time Markov process $\{(\boldsymbol{N}(t),H(t),G(t),\Xi(t)),\ t\geq 0\}$, conditioned on the event $\boldsymbol{N}(t)=\mathbf{0}$, is irreducible on the state space $\{\mathbf{0}\}\times\mathcal{S}$ (cf. also section 2.3.3). Alternatively, it is rather tedious, but straightforward, to verify that the determinant of all but one of the equations (4.17), together with (4.30), is equal to $\Delta=\sigma\Pi_{g=1}^{s}\Pi_{h=1}^{s}\Pi_{\xi=1}^{\Psi_{g,h}^0}\mu_{g,h}^{0,\xi}>0$, implying that this set of equations indeed possesses a unique solution.

We are now ready to give a computational scheme to calculate the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$. The coefficients of the power-series expansions of the state probabilities, and their derivatives with respect to the routing probabilities, can be computed (up to the power M of ρ) according to the following computational scheme:

step 1 : $m:=0$;

step 2 : for all $(k;\boldsymbol{n})$ with $\boldsymbol{n}\neq\mathbf{0}$ and $k+|\boldsymbol{n}|=m$,

(1) for $g\notin\mathcal{N}_{\boldsymbol{n}}^{(0)}$, determine $b_r(k;\boldsymbol{n},h,-g,\xi)$, $h=1,\ldots,s$, $\xi=1,\ldots,\Psi_{g,h}^0$, from (4.17), $r=0,1,\ldots,s^2$;

(2) determine $C_r(k;\boldsymbol{n},g)$, $g\in\mathcal{N}_{\boldsymbol{n}}^{(0)}$, by solving the set of equations (4.21) and determine $b_r(k;\boldsymbol{n},h,-g,\xi)$, $h=1,\ldots,s$, $g\in\mathcal{N}_{\boldsymbol{n}}^{(0)}$, $\xi=1,\ldots,\Psi_{g,h}^0$, from (4.23), in increasing order with respect to $\prec$ (4.28), $r=0,1,\ldots,s^2$;

(3) determine $b_r(k;\boldsymbol{n},h,1,\xi)$, $h=1,\ldots,s$, $\xi=1,\ldots,\Psi_h^0$, according to (4.16), $r=0,1,\ldots,s^2$;

step 3 : determine $C_r(m;\mathbf{0},g)$, $g=1,\ldots,s$, according to (4.21), (4.32), and determine $b_r(m;\mathbf{0},h,-g,\xi)$, $h,g=1,\ldots,s$, $\xi=1,\ldots,\Psi^0_{g,h}$, accor- (4.23), $r=0,1,\ldots,s^2$;

step 4 : $m:=m+1$; if $m\leq M$ then return to *step 2*; otherwise STOP.

General performance measures of the form $\mathrm{E}\{g^{(l)}(\boldsymbol{N},\mathbf{H},\mathbf{G},\Xi)\}$, $l=1,\ldots,L$, and their derivatives with respect to the routing probabilities, can be computed along the same lines as discussed in chapters 2 and 3.

There are various ways to define the derivatives of the routing probabilities with respect to other routing probabilities, $\frac{\partial p_{g,h}}{\partial p_{i,j}}$. Throughout, we will consider routing probability $p_{g,h}$ as function of underlying variables $t_{g,h}$ as follows: for $g,h=1,\ldots,s$,

$$p_{g,h}=\frac{t_{g,h}}{\sum\limits_{k=1}^{s}t_{g,k}}, \tag{4.33}$$

evaluated at $\Sigma_{k=1}^{s}t_{g,k}=1$. Then the derivatives of the routing probabilities are defined as follows: for $g,h,i,j=1,\ldots,s$,

$$\frac{\partial p_{g,h}}{\partial p_{i,j}}:=\left[\frac{\partial p_{g,h}}{\partial t_{i,j}}\right]_{\Sigma_{k=1}^{s}t_{g,k}=1}. \tag{4.34}$$

It is readily verified by applying standard rules for differentiation that: for $g,h,i,j=1,\ldots,s$,

$$\frac{\partial p_{g,h}}{\partial p_{i,j}}=I\{i=g\}\left[I\{j=h\}-p_{g,h}\right]. \tag{4.35}$$

According to the computational scheme, for each $(r,k;\boldsymbol{n})$-combination a set of $|N^{(0)}_{\boldsymbol{n}}|$ linear equations has to be solved (cf. (4.17), (4.21)). Thus, in general, the coefficients of the power series can *not* be determined *fully recursively* under Markovian server routing, as opposed to the special case of cyclic server routing, as elaborated upon in chapter 2.
To characterize whether for given $(r,k;\boldsymbol{n})$, the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$, $h=1,\ldots,s$, $g=-1,1,\ldots,s$, $\xi=1,\ldots,\Psi^0_{g,h}$, are fully recursively solvable, let us reconsider the set of equations (4.17). The states $(r,k;\boldsymbol{n},h,-g,\xi)$ for which the first term at the right-hand side does not vanish are exactly those for which the server has just skipped Q_g which was empty (i.e. $n_g=0$) upon arrival of the server at that queue. Thus, *for given* $(r,k;\boldsymbol{n})$, the set of states $(r,k;\boldsymbol{n},h,-g,\xi)$ which can not be completely ordered are those in which the server is switching between the empty queues. Hence, as long as no arrivals occur at one of the empty queues, the server can keep on switching between these empty queues, provided the routing probabilities corresponding to these switches are strictly positive. For the states $(r,k;\boldsymbol{n},h,-g,\xi)$ with $n_g=0$, the indicator function

in (4.17) does not vanish, so that the coefficients $b_r(k;\boldsymbol{n},h,-g,\xi)$ can not be solved recursively according to (4.17).
This also explains why in the special case of cyclic server routing the states can be computed fully recursively for $\boldsymbol{n} \neq \boldsymbol{0}$ (as elaborated upon in chapter 3). To see this, for $\boldsymbol{n} \neq \boldsymbol{0}$, there exists some index i such that $n_i > 0$, so that the server can not skip Q_i and hence, can not be moving around as long as no arrivals occur. From (4.17) it follows that under cyclic polling with $p_{g,g+1} = 1$, $g = 1,\ldots,s$, the coefficients $b_r(k;\boldsymbol{n},g+1,-g,\xi)$ can be determined recursively in the order $i, i+1,\ldots,s,1,\ldots,i-1$ with respect to g.

For the case in which some or all switch-over times are 0 a.s., some straightforward modifications of the balance equations and of the computational scheme have to be made.

As indicated in section 2.3.4, it is not easy to give bounds for the accuracy of the computations with the PSA. However, for the present polling model with Markovian server routing a rough indication of the accuracy can be obtained from the PCL, i.e. an exact expression for a specific weighted sum of the mean waiting times at the queues. The accuracy of the computations with the PSA can be rougly estimated by computing this specific weighted sum on the basis of the computed mean waiting times and comparing this value to the exact value of the right-hand side of the PCL. For polling models with Markovian server routing, the following relations have been derived in [48], [171]:

$$\begin{aligned}\sum_{i=1}^{s}\rho_i \mathrm{E}W_i &= \tfrac{\rho^2}{2(1-\rho)}\sum_{i=1}^{s}a_i\beta_i^{(2)} + \tfrac{\rho}{2\sigma}\sum_{i=1}^{s}\omega_i\sum_{j=1}^{s}p_{i,j}\sigma_{i,j}^{(2)} \\ &+ \tfrac{1}{\sigma}\sum_{i=1}^{s}\omega_i\sum_{j=1}^{s}p_{i,j}\sigma_{i,j}^{(1)}\sum_{k\neq i}\rho_k\mathrm{E}T_{k,i} + \sum_{i=1}^{s}\mathrm{E}M_i,\end{aligned} \tag{4.36}$$

where $T_{i,j}$ is defined as the time elapsed between a departure of the server from Q_j and its last previous departure from Q_i, $i,j = 1,\ldots,s$. The quantity M_i stands for the amount of work in Q_i at a departure epoch of the server from Q_i and hence, depends on the service discipline at Q_i, $i = 1,\ldots,s$. For the present model with Bernoulli service disciplines, $\mathrm{E}M_i$ is related to $\mathrm{E}W_i$ by the following relation (cf. [162]): for $i = 1,\ldots,s$,

$$\mathrm{E}M_i = (1-q_i)\left[\rho_i\lambda_i\frac{1}{\omega_i}\frac{\sigma}{1-\rho}\mathrm{E}W_i + \rho_i^2\frac{1}{\omega_i}\frac{\sigma}{1-\rho}\right], \tag{4.37}$$

so that the PCL for the present model with Bernoulli service disciplines reads as follows:

$$\begin{aligned}&\sum_{i=1}^{s}\rho_i\left[1-\tfrac{a_i(1-q_i)}{\omega_i}\tfrac{\rho\sigma}{1-\rho}\right]\mathrm{E}W_i = \\ &\tfrac{\rho^2}{2(1-\rho)}\sum_{i=1}^{s}a_i\beta_i^{(2)} + \tfrac{\rho}{2\sigma}\sum_{i=1}^{s}\omega_i\sum_{j=1}^{s}p_{i,j}\sigma_{i,j}^{(2)} \\ &+\tfrac{1}{\sigma}\sum_{i=1}^{s}\omega_i\sum_{j=1}^{s}p_{i,j}\sigma_{i,j}^{(1)}\sum_{k\neq i}\rho_k\mathrm{E}T_{k,i} + \tfrac{\sigma}{1-\rho}\sum_{i=1}^{s}\tfrac{\rho_i^2(1-q_i)}{\omega_i}.\end{aligned} \tag{4.38}$$

The unknown quantities $ET_{k,i}$ can be obtained by solving a set of linear equations. Thus, the PCL for Markovian polling is *not* a closed-form expression, and can only be evaluated by solving a set of linear equations, as opposed to the case of cyclic polling, in which the PCL (3.26) gives a closed-form expression for a weighted sum of the mean waiting times at the queues. Still, the PCL (4.38) is very useful for getting an indication of the accuracy of the calculations with the PSA.

4.4 Markovian versus periodic polling

In many polling models the server has no global information about the actual state of the system, and is therefore routed along the queues according to some state-independent (static) routing mechanism. Static routing mechanisms may be classified as *periodic*, when the visit order is prescribed by a fixed service order table, or as *probabilistic*, when the actual visit order is generated by some random mechanism. In this section we make a *comparison* between the performance of polling models under periodic and probabilistic server routing. To this end, we have implemented the PSA for both polling models with Markovian server routing (cf. section 4.3) and for polling models with periodic server routing (cf. [22]).

Due to the degrees of freedom in specifying the relative arrival rates, the service times, the system load, the switch-over times, the service disciplines and the visit orders, we have restricted ourselves to the analysis of a number of specific models, which we believe cover the main characteristics of the variety of models. Moreover, because of the computational complexity of the PSA we have restriced ourselves to models with a rather small number of queues. The characteristics observed for these models contribute to the understanding of the behavior of polling systems. We believe that these insights are also useful for understanding the behavior of models with a large number of queues. The following models have been taken under consideration, covering fully symmetrical models, models with asymmetrical arrival rates and models with asymmetrical switch-over times. For each of these models the offered load has been varied to cover models under light and heavy traffic, and the service discipline has been varied to cover 1-limited and exhaustive service. In addition, the asymmetry in the arrival rates and switch-over times has been varied to cover a fairly broad class of models.

Model I represents *symmetrical* models, and is specified by the following set of system parameters: $s = 3$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$; $\sigma_{i,j}^{(1)} = 0.05$, $i, j = 1, \ldots, s$; all service times and switch-over times are exponentially distributed; $\boldsymbol{a} = (1.00, 1.00, 1.00)$; $\boldsymbol{q} = (q, q, q)$. The quantities ρ and q are still variable.

Model II represents models in which the *arrival rates* are *asymmetrical*, and is specified by the following set of parameters: $s = 3$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$;

$\sigma_{i,j}^{(1)} = 0.05$, $i,j = 1,\ldots,s$; all service times and switch-over times are exponentially distributed; $\boldsymbol{q} = (q,q,q)$; the relative arrival rates are given by $\boldsymbol{a} = (\alpha/(\alpha+2), 1/(\alpha+2), 1/(\alpha+2))$, so that the ratios between the arrival rates are α:1:1. The quantities ρ, α and q are still variable.

Model III represents models in which the *switch-over times* between the queues are *asymmetrical*, and is defined by the following set of system parameters: $s = 3$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$; all service times and switch-over times are exponentially distributed; $\boldsymbol{a} = (1/3, 1/3, 1/3)$; $\boldsymbol{q} = (q,q,q)$; the mean switch-over times are given by $\sigma_{1,1}^{(1)} = \sigma_{2,2}^{(1)} = \sigma_{3,3}^{(1)} = 0.005$; $\sigma_{2,3}^{(1)} = \sigma_{3,2}^{(1)} = 0.25$; $\sigma_{1,2}^{(1)} = \sigma_{2,1}^{(1)} = \sigma_{1,3}^{(1)} = \sigma_{3,1}^{(1)} = \alpha$, so that α (for values $\alpha \geq 0.125$) can be viewed as the mean 'distance' between Q_1 on the one hand and Q_2 and Q_3 on the other hand. The quantities ρ, α and q are still variable.

To make a reasonable comparison between the performance of Markovian and periodic polling models, we associate with each periodic service order table $\boldsymbol{\pi} = (\pi(1),\ldots,\pi(L))$ a unique Markovian counterpart in which the matrix of routing probabilities $\mathbf{P} = (p_{i,j})$ is defined by: for $i,j = 1,\ldots,s$,

$$p_{i,j} := \frac{\sum_{k=1}^{L} I\left\{\pi_k = i;\ \pi_{(k \bmod L)+1} = j\right\}}{\sum_{k=1}^{L} I\{\pi_k = i\}}, \tag{4.39}$$

i.e. the fraction of times the server moves to Q_j after a departure from Q_i under polling table $\boldsymbol{\pi}$. For instance, if the periodic service order table is given by $\boldsymbol{\pi} = (1,2,1,3)$, then the probabilistic version has routing probabilities $p_{1,2} = p_{1,3} = 0.50$, $p_{2,1} = p_{3,1} = 1.00$. Throughout, the Markovian polling model that is related to a periodic polling model through (4.39) is referred to as the *Markovian counterpart* of the periodic polling model.

In the remainder of this section we show some of the numerical results that we have gathered to compare the performance of polling systems in which the service order is guided by a polling table $\boldsymbol{\pi}$ and their Markovian counterpart. For given routing matrix $\mathbf{P}$, the performance measure considered here is

$$C(\mathbf{P}) := \sum_{i=1}^{s} \rho_i E W_i, \tag{4.40}$$

i.e. the mean total amount of waiting work in the system.
In the numerical examples considered here, the offered load to the system is either $\rho = 0.3$ (representing lightly-loaded systems) or $\rho = 0.8$ (representing heavily-loaded systems). The number of terms of the power series that has been computed is equal to $M = 40$, and the estimated error in the computations (cf. section 2.3.4) is typically less than 0.001.

Let us first consider symmetrical models under symmetrical visit orders, i.e. in which the routing is statistically the same for all queues. To this end, we have computed the system performance (4.40) for model I under a number of symmetrical routing orders with periodic polling (indicated by P) and with their Markovian counterparts (indicated by M). Table 4.1 below shows the results for $q = 0.00$ (1-limited service) and $q = 1.00$ (exhaustive service), and for $\rho = 0.3$ and $\rho = 0.8$.

		$\rho = 0.3$		$\rho = 0.8$	
routing	P/M	$q = 0.00$	$q = 1.00$	$q = 0.00$	$q = 1.00$
123	P, M	0.18	0.17	4.50	3.44
112233	P	0.19	0.18	4.55	3.62
	M	0.20	0.19	4.75	3.64
111222333	P	0.20	0.20	4.63	3.80
	M	0.22	0.21	5.00	3.84
111122223333	P	0.22	0.21	4.74	3.99
	M	0.24	0.23	5.25	4.04

Table 4.1: Performance under symmetrical routing mechanisms; Model I.

Table 4.1 suggests that in symmetrical models the mean total amount of waiting work in the system and hence, the mean waiting times (which are the same for all queues), are smaller in the case of periodic polling than under the corresponding Markovian server routing in all considered cases. However, one may also observe that the differences are rather small.

A comparison of the system performance for the various routing orders considered here shows that the performance of the system is closely related to the *time spacing* of the visits. When the visits are more homogeneously spaced in time, the system performance is likely to be improved. This observation is based on the following intuitive arguments. Define the cycle time C_i of Q_i as the time interval between two successive departures of the server from Q_i. Under Bernoulli service $\mathrm{E}W_i$ (approximately) relates to the first two moments of C_i according to the relation (cf. (3.48)): for $i = 1, \ldots, s$,

$$\mathrm{E}W_i \approx \frac{(1-\rho+\rho_i) - q_i\rho_i(2-\rho)}{1-\rho\,[1+\sigma a_i(1-q_i)/\omega_i]} \frac{\mathrm{E}C_i^2}{2\mathrm{E}C_i}, \tag{4.41}$$

where $\mathrm{E}C_i = \sigma/(\omega_i(1-\rho))$, $i = 1, \ldots, s$, independent of the visit order. Hence, for a given set of system parameters and relative visit frequencies, the mean waiting time at Q_i increases with increasing 'irregularity' of the cycle times, represented by $\mathrm{E}C_i^2$. Now, the *spacing* of the *visit order* will generally be more regular under periodic polling than under Markovian polling. Moreover, a more homogeneous spacing of the visit order is likely to lead to a more homogeneous time spacing of the visit, leading to smaller values of $\mathrm{E}C_i^2$, $i = 1, \ldots, s$. This implies that a more regular visit order is likely to lead to a decrease of the

mean waiting times. These intuitive arguments support the observation in Table 4.1 that the system performance under periodic polling is better than under Markovian server routing.

To investigate whether a similar dominance relation also holds for symmetrical models under asymmetrical server routing, we have computed the mean waiting times $\mathrm{E}\boldsymbol{W} = (\mathrm{E}W_1, \mathrm{E}W_2, \mathrm{E}W_3)$ for model I for a number of asymmetrical service orders, specified by $\boldsymbol{\pi} = (1,2,1,3)$ and $\boldsymbol{\pi} = (1,2,3,1,3,2)$. Table 4.2 shows the results for $q = 0.00$ and $q = 1.00$ and for $\rho = 0.3$ and $\rho = 0.8$.

			$\rho = 0.3$		$\rho = 0.8$	
routing	q	P/M	$\mathrm{E}\boldsymbol{W}$	$C(\cdot)$	$\mathrm{E}\boldsymbol{W}$	$C(\cdot)$
1213	0.00	P	(0.48,0.66,0.66)	0.18	(2.35,7.92,7.92)	4.85
	0.00	M	(0.48,0.74,0.74)	0.20	(2.31,8.29,8.29)	5.04
	1.00	P	(0.48,0.60,0.60)	0.17	(3.03,4.98,4.98)	3.46
	1.00	M	(0.48,0.67,0.67)	0.18	(3.09,5.20,5.20)	3.60
123132	0.00	P	(0.58,0.60,0.60)	0.18	(5.60,5.66,5.66)	4.51
	0.00	M	(0.61,0.61,0.61)	0.18	(5.73,5.73,5.73)	4.58
	1.00	P	(0.54,0.57,0.57)	0.17	(3.92,4.55,4.55)	3.47
	1.00	M	(0.57,0.57,0.57)	0.17	(4.38,4.38,4.38)	3.50

Table 4.2: Performance under asymmetrical routing mechanisms; Model I.

Table 4.2 suggests that in the case of periodic polling the mean amount of work is still smaller than under Markovian polling. Yet, a similar stochastic dominance relation is *not* generally valid for the *individual* mean waiting times. This observation is supported by the following intuitive arguments. Let us reconsider the model with $\boldsymbol{\pi} = (1,2,3,1,3,2)$ with $\rho = 0.8$ in Table 4.2. In that case, the polling order suggests that the visits to Q_1 are more homogeneously spaced than the visits to Q_2 and Q_3. Accordingly, $\mathrm{E}W_1$ can be expected to be smaller than $\mathrm{E}W_2$ and $\mathrm{E}W_3$, which indeed turns out to be the case. Moreover, one may observe that the stochastic counterpart of this model, having routing probabilities $p_{1,2} = p_{1,3} = p_{2,1} = p_{2,3} = p_{3,1} = p_{3,2} = 0.50$, is symmetric, leading to the same mean waiting times at the queues. One would expect $\mathrm{E}W_1$ to be smaller under periodic polling here, because the visits to Q_1 seem to be more homogeneously spaced in time than under Markovian polling, in which the uncertainty leads to less well-spaced visits to Q_1. As for the mean waiting times at Q_2 and Q_3, there is a *trade-off* between the irregularity in the cycle times caused by the use of probabilistic polling on the one hand, and the irregularity of the cycle times caused by a rather bad spacing of the visits under periodic polling on the other hand. Apparently, the former irregularity is *dominated* by the latter one. This intuitive argument supports the observation that in this example $\mathrm{E}W_2$ and $\mathrm{E}W_3$ are larger under periodic polling than under the corresponding Markovian polling mechanism.

To investigate whether the observations made for symmetrical models also persist for asymmetrical models, we consider the performance of the system for both periodic and Markovian service order for a model with varying asymmetry in the arrival rates. We have computed the mean waiting times in model II for a number of values of relative arrival rates. Table 4.3 shows the results where the ratios between the arrival rates are 1:10:10 and 10:1:1, for $q = 1.00$ and $\rho = 0.3$ and $\rho = 0.8$.

			$\rho = 0.3$		$\rho = 0.8$	
routing	ratios	P/M	$\mathrm{E}\boldsymbol{W}$	$C(\cdot)$	$\mathrm{E}\boldsymbol{W}$	$C(\cdot)$
1213	1:10:10	P	(0.51,0.58,0.58)	0.17	(4.02, 4.35, 4.35)	3.39
	1:10:10	M	(0.52,0.65,0.65)	0.18	(4.14, 4.59,4.59)	3.55
	10: 1: 1	P	(0.48,0.72,0.72)	0.19	(2.85,10.78,10.78)	6.51
	10: 1: 1	M	(0.48,0.79,0.79)	0.21	(2.86,11.00,11.00)	6.63
123132	1:10:10	P	(0.59,0.56,0.56)	0.17	(4.66, 4.37, 4.37)	3.57
	1:10:10	M	(0.66,0.56,0.56)	0.18	(7.59, 4.17, 4.17)	4.25
	10: 1: 1	P	(0.51,0.68,0.68)	0.19	(3.14, 9.43, 9.43)	5.87
	10: 1: 1	M	(0.53,0.70,0.70)	0.19	(3.08,10.01,10.01)	6.16

Table 4.3: Performance for asymmetrical routing mechanisms; Model II.

The results in Table 4.3 confirm the observation that the mean amount of work in the system is smaller for periodic polling in all considered cases, but that in a number of cases some of the individual mean waiting times are smaller under Markovian polling.

4.5 Optimization

In this section we consider the following optimization problem:

$$\min_{\mathbf{P} \in \mathcal{M}_s} C(\mathbf{P}), \tag{4.42}$$

where $C(\mathbf{P})$ is defined in (4.40) and $\mathcal{M}_s$ is defined as the set of irreducible stochastic $s \times s$ matrices. In words, the problem is to find combinations of routing probabilities that minimize the mean amount of waiting work in the system. Optimal routing matrices are denoted by $\mathbf{P}^*$.
In a general parameter setting no explicit expressions for the cost function are available and hence, the optimization problem is not exactly solvable.

Boxma et al. [45] consider a similar problem of heuristically obtaining periodic polling tables that minimize the mean amount of work in the system. They propose to combine explicit square-root formulas for optimal relative visit frequencies in random polling models with the Golden Ratio procedure (cf. [98])

for the spacing of the visits. However, this approach relies on the assumption that the switch-over times depend *only* on the queue that is being switched *to*, and is independent of the queue that has just been visited. This assumption is quite restrictive, e.g. when switch-over times represent physical movement from one place to another. Yet, when this assumption is dropped, the problem of finding an optimal visit order for *given* relative visit frequencies can be formulated as a Travelling Salesman Problem (TSP), which is known to be NP-hard.

The optimization problem (4.42) can, in principle, be solved numerically by combining a numerical algorithm for the evaluation of the cost function (4.40) with some standard procedure for non-linear (constrained) optimization. However, the dimension of the optimization problem grows quadratically in the number of queues, so that in practice this approach is restricted to models with a rather small number of queues. It should be noted that in the special case in which all queues are served exhaustively, the cost function (4.40) can be directly obtained via the PCL (4.38), requiring the solution of a relatively small set of equations. However, in case at least one queue is non-exhaustively served, the PCL is no longer applicable to evaluate the cost function (4.40). Moreover, the PCL can not be used to determine more detailed performance measures like the individual mean waiting times at the queues. In those situations, the computations may be based on the use of the PSA, requiring considerably more computational effort. To find optimal routing matrices, we have computed the cost function (4.40), plus its derivatives with respect to the routing probabilities, in combination with the conjugate gradient method for non-linear optimization with linear constraints (cf. [141]).
We reemphasize the enormous complexity of the TSP-like optimization problem in a general parameter setting. Therefore, we restrict ourselves to obtain some qualitative, instead of quantitative, properties of optimal combinations of routing probabilities. The results presented here should be viewed in this perspective.

The remainder of this section is organized as follows. In section 4.5.1 we discuss properties of optimal routing matrices in the case of fully symmetrical models. In section 4.5.2 we investigate optimal combinations of routing probabilities in the case of some asymmetrical models.

4.5.1 Symmetrical systems

For fully symmetrical models it is shown in [125] that each cyclic service order, which is contained in the class of Markovian service orders, solves the optimization problem (4.42). Thus, for such models with (symmetrical) Bernoulli schedule $\boldsymbol{q} = (q, \ldots, q)$, $0 \leq q \leq 1$,

$$\mathbf{P}^* = \hat{\mathbf{P}}, \tag{4.43}$$

where $\hat{\mathbf{P}} = (\hat{p}_{i,j}) \in \mathcal{M}_s$ with $\hat{p}_{i,j} \in \{0,1\}$, $i,j = 1,\ldots,s$. Note that there are $(s-1)!$ alternative *local optima*, each of which uniquely corresponds to a specific cyclic visit order.

Let us consider the question whether these optima are stable, i.e. whether the optimal cyclic server routing orders remain optimal when the system parameters are slightly perturbed. To this end, it should be noted that the derivatives of the cost function (4.42) with respect to each of the routing probabilities may provide useful information about the character of the optimal routing probabilities. Namely, when all derivatives are equal to 0 in the optimum, the optimal schedule will be an 'interior optimum' which may become suboptimal for slight changes in one of the system parameters. On the other hand, 'boundary optima' with non-zero derivatives at the optimum remain (locally) optimal for slight modifications of the parameters. To study the character of the optima, we have applied the PSA to compute the cost function (4.40) and the derivatives with respect to the routing probabilities for a set of symmetrical models, each of which giving similar outcomes. For a typical example, consider Model I (introduced in section 4.4). Table 4.4 shows the matrix of derivatives of the cost function (4.42) with respect to the routing probabilities at $\mathbf{P}^* = (p^*_{i,j})$, with $p^*_{1,2} = p^*_{2,3} = p^*_{3,1} = 1.00$, for $\rho = 0.3$ and 0.8 and for $q = 0.00$, 0.50 and 1.00.

	$q = 0.00$	$q = 0.50$	$q = 1.00$
$\rho = 0.3$	0.007 0.000 0.007	0.007 0.000 0.007	0.007 0.000 0.007
	0.007 0.007 0.000	0.007 0.007 0.000	0.007 0.007 0.000
	0.000 0.007 0.007	0.000 0.007 0.007	0.000 0.007 0.007
$\rho = 0.8$	0.030 0.000 0.030	0.028 0.000 0.028	0.025 0.000 0.025
	0.030 0.030 0.000	0.028 0.028 0.000	0.025 0.025 0.000
	0.000 0.030 0.030	0.000 0.028 0.028	0.000 0.025 0.025

Table 4.4: Derivatives $\partial C(\mathbf{P})/\partial p_{i,j}$ at $\mathbf{P} = \mathbf{P}^*$ in a symmetrical model.

Table 4.4 suggests that the optimal cyclic visit orders are *stable optima*. To illustrate this, let us consider the case q=0.50, $\rho = 0.8$. Consider the routing probabilities after departing from Q_1, which are under the present cyclic schedule equal to $p^*_{1,2} = 1.00$, $p^*_{1,1} = p^*_{1,3} = 0.00$. To obtain an alternative triple of routing probabilities, either $p_{1,1}$ or $p_{1,3}$, or both, must be increased, and $p_{1,2}$ will have to be decreased. Now, because the derivatives of the cost function with respect to $p_{1,1}, p_{1,2}$ and $p_{1,3}$ at $\mathbf{P}^*$ are given by 0.028, 0.000 and 0.028, respectively, a (small) increase in either $p_{1,1}$ or $p_{1,3}$ together with a (small) decrease of $p_{1,2}$ will lead to an increase of the cost function. Under the assumption that the derivatives of the cost function with respect to continuous system parameters at the cyclic optimum are continuous, slight modifications of these system parameters do not cause the cyclic optimum to become suboptimal.

The fact that in Table 4.4 the derivatives with respect to $p_{1,2}$, $p_{2,3}$ and $p_{3,1}$ are 0.000 is due to the definition of the derivatives of the routing probabilities in (4.33)-(4.35), which even implies that these derivatives are *exactly* equal to zero.

The observation that the cyclic optimum is stable suggests the existence of a non-empty *attraction region* for the cyclic optimum for nearly-symmetrical models. That is, there is some set of 'nearly symmetrical' models for which the optimal Markovian server routing is cyclic. In the next subsection we will present some numerical experiments that support this observation. Yet, exact expressions for this region are unknown, so that the observation is only useful for providing a qualitative, rather than a quantitative, insight into optimal combinations of routing probabilities.

4.5.2 Asymmetrical systems

When the model is asymmetrical, it is clear that the cyclic server routing is no longer generally optimal within the class of Markovian server routings. However, for asymmetrical models the optimization problem is not exactly solvable, and numerical procedures are needed to find optimal combinations of routing probabilities. In a number of examples discussed here, it is assumed that all queues are served exhaustively. This assumption is based on results obtained by Liu et al. [125], who have shown that in order to minimize the cost function (4.40), all queues should be served exhaustively. Recall that in those cases the cost function (4.40), and the optimal routing matrices, can be obtained relatively quickly via (4.38). Yet, in some applications exhaustive service may not be implemented or technically infeasible. In those situations, the cost function (4.40), and the optimal routing matrices, have been obtained on the basis of the PSA.
As for the accuracy of the computations, in the case of exhaustive service at all queues the cost function (4.40) has been accurately calculated from the PCL (4.38), with typical errors less than 10^{-12}. In the cases with non-exhaustive service disciplines the cost function has been evaluated by means of the PSA, where typically 40 or 50 terms of the power series have been computed and the estimated errors are typically less than 0.001. The optimization procedure is based on a grid size 0.001.
Numerical experience has taught us that the cost function (4.40) as function of the routing matrices generally has a number of *local optima*, similar to the case of fully symmetrical models. This difficulty, which is very common in non-linear optimization, is tackled by running the optimization procedure with a number of different initial routing matrices.

To study characteristics of optimal routing matrices for a broad class of Markovian polling models, we have computed the optimal routing probabilities for a wide variety of the parameter settings for models II and III (model I occurring as a special case), covering a diversity of models. In this section we present some of the numerical results. We emphasize that this numerical study does

not aim to give a full characterization of optimal schedules, but is meant to give some useful insights, which contribute to the understanding of the characteristics of optimal routing matrices.

Influence of asymmetry in the arrival rates
To investigate the influence of the asymmetry in the arrival process on $\mathbf{P}^*$, we have computed an optimal routing matrix for model II (cf. section 4.4) for $q = 1.00$ and in which the ratios between the arrival rates are given by α:1:1. Tables 4.5 and 4.6 show an optimal routing matrix $\mathbf{P}^*$ for various values of α for the system under light traffic ($\rho = 0.3$) and heavy traffic ($\rho = 0.8$), respectively. It should be noted that because in this model Q_2 and Q_3 are stochastically identical, the corresponding routing probabilities are exchangeable.

	$\alpha = 0.001$	$\alpha = 0.10$	$\alpha = 0.25$	$\alpha = 0.50$
$\mathbf{P}^*$	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00
	0.04 0.00 0.96	0.48 0.00 0.52	0.81 0.00 0.19	1.00 0.00 0.00
	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00
$C(\cdot)$	0.155	0.162	0.164	0.165

	$\alpha = 4.00$	$\alpha = 10.00$	$\alpha = 100.00$	$\alpha = 1000.00$
$\mathbf{P}^*$	0.00 0.00 1.00	0.00 0.44 0.56	0.00 0.50 0.50	0.92 0.04 0.04
	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00
	0.00 1.00 0.00	0.80 0.20 0.00	1.00 0.00 0.00	1.00 0.00 0.00
$C(\cdot)$	0.163	0.160	0.152	0.146

Table 4.5: Optimal routing probabilities for model II; $\rho = 0.3$.

	$\alpha = 0.001$	$\alpha = 0.10$	$\alpha = 0.25$	$\alpha = 0.50$
$\mathbf{P}^*$	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00
	0.05 0.00 0.95	0.58 0.00 0.42	0.94 0.00 0.06	1.00 0.00 0.00
	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00
$C(\cdot)$	3.347	3.402	3.422	3.434

	$\alpha = 10.00$	$\alpha = 25.00$	$\alpha = 50.00$	$\alpha = 100.00$
$\mathbf{P}^*$	0.00 0.00 1.00	0.00 0.40 0.60	0.00 0.50 0.50	0.22 0.39 0.39
	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00
	0.00 1.00 0.00	0.66 0.34 0.00	1.00 0.00 0.00	1.00 0.00 0.00
$C(\cdot)$	3.350	3.311	3.287	3.274

Table 4.6: Optimal routing probabilities for model II; $\rho = 0.8$.

The results displayed in Tables 4.5 and 4.6 reveal some characteristics of optimal routing matrices.
First, we observe the surprisingly large fraction of the routing probabilities that are equal to 0.00 or 1.00, indicating that the optimal routing decisions have a tendency towards *deterministic* routing. This observation is supported by the observation in section 4.4 that the cycle times under deterministic routing are more regular than under probabilistic routing decisions, generally leading to a better system performance (cf. (4.41)).
Second, for lightly- and heavily-loaded systems the results suggest that the optimal matrices for different values of α can be divided into a small number of classes, each of which has specific characteristics that can be interpreted easily, providing an insight into the behavior of optimal routing matrices. Each class corresponds to a specific interval of values of α. We will now discuss characteristics of these classes, letting α increase from 0 to infinity.

Let us first consider the limiting case $\alpha \downarrow 0$. In that case the arrival rate at Q_1 is negligible compared with the arrival rates at Q_2 and Q_3, so that in the optimum Q_1 will be visited only very seldom. The results in Tables 4.5 and 4.6 suggest that $\mathbf{P}^*$ tends to a limiting routing matrix with $p^*_{1,3} = p^*_{2,3} = p^*_{3,2} = 1.00$. Under this routing matrix, state 1 (corresponding to visits to Q_1) is only a transient state, and the states 2 and 3 form an absorbing set of states.
When α is somewhat increased, starting from 0.00, Q_1 will be visited more frequently, but still less frequently than Q_2 and Q_3. This situation appears to give optimal routing matrices of the form $p^*_{1,3} = p^*_{3,2} = 1.00$, $p^*_{2,1} = r$, and $p^*_{2,3} = 1 - r$, $0 \leq r < 1$, where the value of r increases with increasing value of α. That is, after a departure from Q_1 the server always moves to Q_3 and subsequently, to Q_2. The only random routing decisions are made after departures from Q_2. So, the server visits the queues in cyclic order, typically interceded by a number of switches back and forth between Q_2 and Q_3.
When α is increased to approach 1.00, the arrival rates become of the same order of magnitude. For $\alpha = 1.00$ the system is symmetric, and it is known that in that case the optimal routing is cyclic (cf. [126]). Moreover, Tables 4.5 and 4.6 suggest that the *cyclic* routing is still optimal when α is varied within some interval around $\alpha = 1.00$. This observation supports the conjecture that there is some region 'around' the cyclic optimum $\mathbf{P}^*$ in which $\mathbf{P}^*$ is still optimal (cf. section 4.5.1).
When the value of α is increased even more, Q_1 becomes considerably more heavily loaded than the other queues, so that above some threshold value for α the cyclic visit order is no longer optimal. We then typically observe optimal routing matrices $\mathbf{P}^*$ of the form $p^*_{2,1} = 1.00$, $p^*_{1,2} = r_1$, $p^*_{1,3} = 1 - r_1$, $p^*_{3,1} = r_2$, $p^*_{3,2} = 1 - r_2$ $(0 < r_1, r_2 < 1)$. Under this type of service orders, Q_1 is implicitly given higher priority than the other queues. This is because after most visits to either Q_2 or Q_3 the next queue to be served is Q_1. The only exception is when Q_2 is visited after a visit to Q_3. In those cases Q_1 will be immediately visited afterwards. This type of routing matrix may be seen as an *intermediate* between the cyclic server routing (for smaller values of α) and

another type of polling order which occurs when α is increased further. In the latter case Q_1 dominates the system in such a strong way that Q_1 is always visited immediately after a visit to one of the other queues, so that the optimum $\mathbf{P}^*$ is typically of the form $p^*_{1,2} = p^*_{1,3} = 0.50$, $p^*_{2,1} = p^*_{3,1} = 1.00$. This type of routing matrix may be seen as a stochastic counterpart of the periodic *star-type* polling.
Finally, when α is increased even further Q_1 becomes so relatively heavily loaded that switches from Q_1 to itself (throughout referred to as *self transitions*) become optimal, while Q_1 is always visited immediately after a visit to one of the other queues. When α approaches infinity, the optimal routing matrix tends to the routing matrix $\mathbf{P}^*$ with $p^*_{1,1} = p^*_{2,1} = p^*_{3,1} = 1.00$ and zeros elsewhere.

Influence of the asymmetry in the switch-over times
To investigate the influence of the switch-over times on the optimal routing matrix, we have computed optimal probabilities for a variety of models which are contained in the class described in model III. For these models the ratios between the arrival rates are equal, and mean switch-over times are given by $\sigma^{(1)}_{1,1} = \sigma^{(1)}_{2,2} = \sigma^{(1)}_{3,3} = 0.005$; $\sigma^{(1)}_{1,2} = \sigma^{(1)}_{2,1} = \sigma^{(1)}_{1,3} = \sigma^{(1)}_{3,1} = \alpha$; $\sigma^{(1)}_{2,3} = \sigma^{(1)}_{3,2} = 0.25$. Note that, for $\alpha \geq 0.125$, the parameter α can basically be viewed as the mean 'distance' between Q_1 on the one hand and Q_2 and Q_3 on the other hand. Tables 4.7 and 4.8 show the results for various values of α, $q = 1.00$, and for ρ=0.3 and 0.8, respectively.

	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.10$	$\alpha = 0.25$
$\mathbf{P}^*$	0.00 0.50 0.50	0.00 0.50 0.50	0.00 0.50 0.50	0.00 0.00 1.00
	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00
	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00	0.00 1.00 0.00
$C(\cdot)$	0.130	0.139	0.236	0.311

	$\alpha = 0.50$	$\alpha = 1.00$	$\alpha = 2.50$	$\alpha = 10.00$
$\mathbf{P}^*$	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00
	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00
	0.95 0.05 0.00	0.54 0.46 0.00	0.25 0.75 0.00	0.07 0.93 0.00
$C(\cdot)$	0.437	0.686	1.412	5.019

Table 4.7: Optimal routing probabilities for model III; $\rho = 0.3$.

The results in Tables 4.7 and 4.8 reveal some properties of the character of optimal routing matrices. Similar to the case of varying the relative arrival rates discussed above, we observe again that the optimal routing decisions have a tendency towards *deterministic* routing (cf. the discussion of the results in Tables 4.5 and 4.6). Moreover, Tables 4.7 and 4.8 indicate that the optimal routing matrices for varying values of α can be divided into a number of types

	$\alpha = 0.001$	$\alpha = 0.01$	$\alpha = 0.10$	$\alpha = 0.25$
$\mathbf{P}^*$	0.00 0.50 0.50	0.00 0.50 0.50	0.00 0.00 1.00	0.00 0.00 1.00
	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00	1.00 0.00 0.00
	1.00 0.00 0.00	1.00 0.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00
$C(\cdot)$	3.208	3.280	3.933	4.400

	$\alpha = 0.50$	$\alpha = 1.00$	$\alpha = 2.50$	$\alpha = 10.00$
$\mathbf{P}^*$	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00	0.00 1.00 0.00
	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00
	1.00 0.00 0.00	0.70 0.30 0.00	0.41 0.59 0.00	0.19 0.81 0.00
$C(\cdot)$	5.213	6.805	11.231	32.129

Table 4.8: Optimal routing probabilities for model III; $\rho = 0.8$.

of routing matrices. We will briefly discuss characteristics of each of these classes.

When α is small the optimal routing matrix routes the server to Q_1 (with probability 1.00) after a visit to one of the other queues. In this way, the relatively long 'journey' between Q_2 and Q_3 is avoided, and Q_1 serves as a 'bridge' between these queues.

When α approaches 0.25 the system becomes symmetrical and the cyclic visit order becomes optimal. Again it is observed that this cyclic optimum remains optimal for slight perturbations in the switch-over times.

When α becomes considerably larger than the switch-over times between Q_2 and Q_3, Q_1 basically becomes relatively 'isolated' from Q_2 and Q_3, or equivalently, Q_2 and Q_3 may be viewed as relatively 'clustered'. The optimal routing matrices $\mathbf{P}^*$ appear to have a specific structure of the form $p^*_{1,2} = p^*_{2,3} = 1.00$ and $p^*_{3,1} = r$, $p^*_{3,2} = 1 - r$ $(0 < r < 1)$, where r decreases with increasing α. This specific structure can be interpreted as follows. After having emptied Q_1 the server always moves towards Q_3, and after a visit to Q_3 the server moves to Q_2 with probability 1.00. Then the server keeps on alternating between Q_2 and Q_3 before making the relatively long trip to Q_1. The latter implies that in this way one avoids making two successive relatively long journeys without having visited both queues in the cluster of Q_2 and Q_3.

Influence of the service disciplines

In the cases considered so far it is assumed that the queues are served exhaustively. We will now study the influence of the service discipline on optimal routing matrices. To this end, consider the case $\alpha = 1.00$ for Model III (cf. also Tables 4.7 and 4.8). Recall that in this case Q_1 is relatively 'isolated' from Q_2 and Q_3. We study the influence of the service discipline at Q_1 on the optimal routing matrices. To this end, we have computed the optimal routing matrices for various Bernoulli service policies with parameter $q_1 = q$ $(0 \leq q \leq 1)$. The

service discipline at Q_2 and Q_3 is assumed to be exhaustive. Tables 4.9 and 4.10 below show optimal routing matrices for q=0.00, 0.50 and 1.00, and for $\rho = 0.3$ and ρ=0.8, respectively.

	$q = 0.00$	$q = 0.50$	$q = 1.00$
$\mathbf{P}^*$	0.96 0.04 0.00	0.95 0.05 0.00	0.00 1.00 0.00
	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00
	0.56 0.44 0.00	0.55 0.45 0.00	0.54 0.46 0.00
$C(\cdot)$	0.696	0.692	0.686

Table 4.9: Optimal routing matrices for different service disciplines; $\rho = 0.3$.

	$q = 0.00$	$q = 0.50$	$q = 1.00$
$\mathbf{P}^*$	0.97 0.03 0.00	0.95 0.05 0.00	0.00 1.00 0.00
	0.00 0.00 1.00	0.00 0.00 1.00	0.00 0.00 1.00
	0.77 0.23 0.00	0.76 0.24 0.00	0.70 0.30 0.00
$C(\cdot)$	7.301	7.163	6.805

Table 4.10: Optimal routing matrices for different service disciplines; $\rho = 0.8$.

The results in Table 4.9 indicate that the service discipline may have a considerable impact on the optimal routing matrices. In particular, we observe a striking difference in the optimal routing probabilities between exhaustive service on the one hand and non-exhaustive service on the other hand. We observe that in the case of non-exhaustive service (i.e. $q < 1$) self transitions occur frequently here, whereas for exhaustive service similar self transitions occur with probability 0. To give an intuitive argument for this observation, recall that it is shown in [126] that all queues should be served exhaustively to minimize the cost function (4.40). Consider the case $q < 1$, i.e. Q_1 is served non-exhaustively, so that after a visit of the server at Q_1 there may be customers present at Q_1. Note that the switch-over times needed by the server for a self transition (with mean 0.005) are negligible compared with the switch-over times between different queues (with means 0.25 or 1.00). Hence, the server can almost immediately return to Q_1 to check whether there is another customer waiting at that queue. If so, the next customer at Q_1 will be served, and if not so, the 'cost' of this 'unnecessary' travel from Q_1 to itself is very small. This argument intuitively explains why for non-exhaustive service at Q_1 self transitions occur with large probability (typically ≥ 0.9). In this way, Q_1 is served 'nearly-exhaustively'. Obviously, in the case of (fully) exhaustive service, self transition from Q_1 would probably not make much sense, because no customers are present at a departure instant of the server at Q_1.

The above-mentioned considerations indicate that when self transitions can be made instantaneously (i.e. $\sigma_{i,i}^{(1)} = 0,\ i = 1,\ldots,s$), then under exhaustive service at Q_i, the cost function does not depend on $p_{i,i}$, provided $p_{i,i} < 1$. Moreover, for systems with $\sigma_{i,i}^{(1)} = 0,\ q_i = 0,\ i = 1,\ldots,s$, the service discipline at Q_i can basically be viewed as a *Bernoulli* service discipline with parameter $\tilde{q}_i = p_{i,i},\ i = 1,\ldots,s$ (cf. also [48]). In this way, the problem of finding optimal Bernoulli parameters $\tilde{\boldsymbol{q}} = (\tilde{q}_1,\ldots,\tilde{q}_s)$ in cyclic polling models (discussed extensively in chapter 3) occurs as a special case by putting the additional restriction $p_{i,i} + p_{i,i+1} = 1$. The only difference here is that $\tilde{q}_i$ should be strictly smaller than 1 (to guarantee the irreducibility of the Markov chain D, cf. section 4.2), while $\tilde{q}_i = 1$ is also allowed in the optimization problem discussed in chapter 3.

Guidelines for constructing optimal routing matrices
Based on the results presented in Tables 4.5 to 4.10, we will now give some general ideas that may be useful for heuristically constructing routing matrices for larger systems. We reemphasize that these ideas only aim to give some insight into the qualitative, rather than the quantitative, behavior of optimal routing matrices, and should be viewed in that perspective.

Suppose the distance structure is such that the queues $Q_1,\ldots,Q_s$ can somehow be partitioned into a relatively small number of *clusters* of queues, $C_1,\ldots,C_m$, $m < s$, in such a way that the mean switch-over times between queues within the same cluster are considerably smaller than the distances between queues in different clusters. In this perspective, each of these clusters can be viewed as a *super queue*. The numerical results in Tables 4.7 and 4.8 suggest that in each cluster C_k there is a 'front door' queue C_k^F such that the server can 'enter' cluster C_k only through a visit at queue C_k^F and not through a visit at another queue in C_k. This suggestion implies that for (nearly-) optimal routing matrices we have $p_{i,j}^* = 0$ if $Q_i \notin C_k,\ Q_j \in C_k$ and $Q_j \neq C_k^F,\ k = 1,\ldots,m$. Similarly, each cluster C_k seems to have a 'back door' queue C_k^B such that the server can only depart from cluster C_k through $C_k^B,\ k = 1,\ldots,m$, i.e. $p_{i,j}^* = 0$ if $Q_i \in C_k,\ Q_j \notin C_k$ and $Q_i \neq C_k^B,\ k = 1,\ldots,m$.
Moreover, one may expect that the optimal routing probabilities within each cluster C_k will be such that after the server has entered C_k (through an arrival at C_k^F) *all* queues within that cluster are certainly visited at least once during the visit of the server to that cluster. The problem of determining optimal routing probabilities between the different clusters, being $p_{i,j}^*,\ Q_i = C_k^B,\ Q_j = C_l^F,\ k,l = 1,\ldots,m$, is roughly similar to the problem of determining optimal routing matrices for systems with $m < s$ (super) queues $\tilde{Q}_1,\ldots,\tilde{Q}_m$, where the parameters of super queue C_k can be determined by *aggregating* over the parameters of the queues in C_k in a straightforward manner. This observation suggests a *hierarchical* procedure for obtaining optimal routing matrices for larger systems.

As an illustration of the validity of the above-mentioned guidelines, we con-

sider the model with the following combination of system parameters: $s = 4$; $\boldsymbol{a} = (1.00, 1.00, 1.00, 1.00)$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00, 1.00)$; all service times and switch-over times are exponentially distributed; $\boldsymbol{q} = (1.00, 1.00, 1.00, 1.00)$; $\sigma_{1,j}^{(1)} = \sigma_{j,1}^{(1)} = \alpha$, for $j = 2, 3, 4$; $\sigma_{i,j}^{(1)} = 0.05$ in all other cases. Note that for values of α large enough, the queues can be basically partitioned into clusters $C_1 = \{Q_1\}$ and $C_2 = \{Q_2, Q_3, Q_4\}$. Table 4.11 shows optimal routing matrices for α=0.10, 0.25 and 5.00, and for ρ=0.3. Table 4.12 shows the results for $\rho = 0.8$.

	$\alpha = 0.10$	$\alpha = 0.25$	$\alpha = 5.00$
$\mathbf{P}^*$	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00
	0.00 0.00 1.00 0.00	0.00 0.00 1.00 0.00	0.00 0.00 1.00 0.00
	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00	0.00 0.32 0.00 0.68
	1.00 0.00 0.00 0.00	0.13 0.87 0.00 0.00	0.04 0.96 0.00 0.00
$C(\cdot)$	0.201	0.608	2.404

Table 4.11: Optimal routing probabilities; $\rho = 0.3$.

	$\alpha = 0.10$	$\alpha = 0.25$	$\alpha = 5.00$
$\mathbf{P}^*$	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00
	0.00 0.00 1.00 0.00	0.00 0.00 1.00 0.00	0.00 0.00 1.00 0.00
	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00
	1.00 0.00 0.00 0.00	0.24 0.76 0.00 0.00	0.10 0.90 0.00 0.00
$C(\cdot)$	3.713	6.222	16.347

Table 4.12: Optimal routing probabilities; $\rho = 0.8$.

The results in Tables 4.11 and 4.12 confirm the characteristics discussed above. Obviously, the queues can be clustered as $C_1 = \{Q_1\}$ and $C_2 = \{Q_2, Q_3, Q_4\}$. We observe that C_2 is only entered through Q_2 (i.e. $C_2^F = Q_2$) and is only departed from at Q_4 (i.e. $C_2^B = Q_4$). In all cases considered here the server moves to C_2 after departing from Q_1, and visits the queues in C_2 a number of times (geometrically distributed with parameter $1 - p_{1,4}^*$) before returning to Q_1. We also observe that once the server has entered C_2 through a visit at Q_2, all queues in C_2 are served at least once during that visit.

As an alternative, consider the model with the same system parameters as the above-discussed model, but with the following mean switch-over times: $\sigma_{i,i}^{(1)} = 0.05$, $i = 1, \ldots, 4$; $\sigma_{i,j}^{(1)}$=1.00 if $i, j \in \{1, 2\}$ or $i, j \in \{3, 4\}$; $\sigma_{i,j}^{(1)} = \alpha$ in all other cases. Note that for $\alpha > 1.00$, the queues can basically be clustered into clusters $C_1 = \{Q_1, Q_2\}$ and $C_2 = \{Q_3, Q_4\}$. Table 4.13 shows the optimal

routing matrices for α=1.00, 10.00 and 50.00 for $\rho = 0.3$. Table 4.14 shows the results for $\rho = 0.8$.

	$\alpha = 1.00$	$\alpha = 10.00$	$\alpha = 50.00$
$\mathbf{P}^*$	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00
	0.00 0.00 1.00 0.00	0.48 0.00 0.52 0.00	0.84 0.00 0.16 0.00
	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00
	1.00 0.00 0.00 0.00	0.52 0.00 0.48 0.00	0.16 0.00 0.84 0.00
$C(\cdot)$	1.071	5.753	26.144

Table 4.13: Optimal routing probabilities; $\rho = 0.3$.

	$\alpha = 1.00$	$\alpha = 10.00$	$\alpha = 50.00$
$\mathbf{P}^*$	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00	0.00 1.00 0.00 0.00
	0.00 0.00 1.00 0.00	0.36 0.00 0.64 0.00	0.72 0.00 0.28 0.00
	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00	0.00 0.00 0.00 1.00
	1.00 0.00 0.00 0.00	0.64 0.00 0.36 0.00	0.28 0.00 0.72 0.00
$C(\cdot)$	10.000	41.090	164.522

Table 4.14: Optimal routing probabilities; $\rho = 0.8$.

Tables 4.13 and 4.14 support the characteristics of the optimal routing matrices discussed in this section. In all cases considered here we have $C_1^F = Q_1$, $C_1^B = Q_2$, $C_2^F = Q_3$ and $C_2^B = Q_4$. The server typically moves from one cluster to the other, interceded by a number of switches back and forth between the queues within the respective clusters.

4.6 Concluding remarks

The guidelines for constructing optimal routing matrices in the previous section are based on the insights obtained by the numerical study presented that section. However, the guidelines are qualitative, and do not yield a heuristic approach to obtain optimal routing matrices. The following idea may be worthwhile to consider for obtaining a quantitative heuristic approach. For a given planar distance structure between the queues (due to the switch-over times), there are various algorithms available for partitioning the set of queues ('points') into a number of subsets ('clusters') of queues (e.g. single-link clustering, complete-link clustering, furthest-neighbor method, cf. e.g. [60]). Each of these algorithms provides a means to define a clustering structure depending on whether the planar distances between certain combinations of queues exceed some threshold value d. For a given clustering algorithm, one may build

a *tree structure* of clusters by successively decreasing the threshold distance d, starting with $d = \infty$ (in which all queues form one cluster) until $d = 0$ (in which each forms a cluster by itself). Such a tree structure suggests an *iterative* approach for heuristically obtaining optimal routing matrices for large systems, with decreasing threshold value d.
By definition of 'iteration', at each step of the iteration at least one couple of clusters, say C_1 and C_2 is united to one cluster $C_{12} := C_1 \cup C_2$. In this way, one should construct (i) a simple heuristic approach to define the 'front door' C_{12}^F and the 'back door' C_{12}^B of cluster C_{12} (defined in section 5), and (ii) a simple heuristic to 'merge' the 'local' routing probabilities for C_1 and C_2 to routing probabilities for C_{12}. As for the first problem, one should probably select C_{12}^B either C_1^B or C_2^B, and a similar approach may be used to determine C_{12}^F. The second problem may be handled by adopting the intra-cluster ('local') routing probabilities. As for the inter-cluster routing probabilities, one may set $p_{i,j} = 0$, $i \in C_1$, $j \in C_2$, unless $Q_i = C_1^B$ and $Q_j = C_2^F$. The routing probabilities between C_1^B, C_1^F, C_2^B, C_2^F can be determined numerically by considering the optimization problem discussed in section 5 for small (two-queue) polling models with Markovian server routing, which can be solved in a similar way as was done in section 4.5. Note that the observed tendency towards deterministic routing may be used here to set certain routing probabilities equal to 1.00.
This iterative algorithm converges when each queue forms a cluster by itself (for small values of the threshold disctance d). It should be noted that the algorithm converges after *at most* s iterations, because (by definition) at each stage at least two queues are united.

We reemphasize the enormous mathematical and numerical complexity of the optimization problem considered here and the ideas should be viewed in that perspective. Although the idea of hierarchical clustering is rather intuitive and may hide some unforseen complications, we believe it is interesting to pursue this idea further in the future.

Chapter 5

Polling systems with a dormant server

5.1 Introduction

In the literature about the analysis of polling models it is typically assumed that at any time the server is either switching or serving. When the server arrives at a queue finding no customers present at that queue, the server immediately starts to move to the next queue. As a consequence, the server keeps on moving around in the system when there are no customers present for some time period. However, in many situations it is more natural for the server to rest when the system is entirely empty. Moreover, in many cases it is likely that considerable improvements of the system performance can be made by allowing the server to rest, especially when the times (or cost) involved in switching from one queue to another are considerable. A polling system in which the server is allowed to rest at a queue when the system is empty will be referred to as a polling system with a dormant server, as opposed to the ordinary non-dormant server polling systems. In general, the server is allowed to rest only at a set of prespecified queues, referred to as the dormant set.
In this chapter we analyze the performance of dormant-server polling models. We investigate the improvements of the system performance that can be made by allowing the server to rest at a queue. In addition, we study the problem of determining a dormant set minimizing an arbitrary weighted sum of the mean waiting times at the queues.

There are sound reasons to consider the option of resting. In many applications the option of resting provides a more natural way of modeling. For instance, an elevator in an empty system may position itself on the main floor, which is the most likely source of new customers, when the system is empty. Alternatively, the option of resting is useful in the dynamic control of traffic lights, where a mainstream is given passage until a waiting vehicle of a crossing stream is de-

tected. Examples can also be found in maintenance environments, where one or more repairmen travel along a number of installations to perform maintenance (cf. [129], [56], [103], [105]). In those cases it makes sense for the repairmen to stay at some home base when no maintenance activities are needed for some time (cf. e.g. chapter 7 of [171]). Other applications of dormant-server polling models may be found in the area of manufacturing, where a machine is used to perform several types of tasks. In those cases switch-over times represent the times needed by the machine to change from one type of operation to another. In the dormant-server case, some mechanism is needed to keep track of customers in the system. In many cases such a mechanism is already available, e.g. in manfacturing environments where some supporting system to schedule the jobs is needed anyhow. In those cases, the option of resting seems to be worth considering. In other cases, a comparison should be made between the benefits that can be made by allowing the server to rest at a queue and the increased overhead involved in keeping track of the customers in the system.

Only a few studies have been devoted to the analysis of polling models with a dormant server. For a two-queue model with exhaustive service at both queues, Eisenberg [76] considers a model with either alternating or strict priority, in which the server rests at a queue as soon as the entire system becomes empty. In a recent study, Eisenberg [79] analyzes a variety of models with an arbitrary number of queues with exhaustive service at all queues. He considers a number of stopping rules, indicating the server behavior when the system is empty ('Stop-immediately', 'Jump-home' and 'Continue-home'), and different starting rules for the server behavior when a new customer arrives when the server is resting ('Resume-cycle' and 'Jump-to-arrival'). For a number of combinations of these rules, he derives expressions for the Laplace-Stieltjes Transform (LST) of the waiting-time distribution at each of the queues. For a model with exhaustive service at all queues and in which the server stops as soon as the system becomes empty, Gupta and Srinivasan [95] develop an efficient method for calculating the mean waiting times at the queues. As a by-product, they derive a pseudo-conservation law (PCL) for this model.
The option of resting arises with the so-called globally-gated service discipline, introduced in [46]. Under this service discipline, during the tour along the queues exactly those customers are served that were present in the system at the beginning of the tour, while the customers which arrive during that tour have to wait until the next cycle. For models with globally-gated service in which the server rests at its home base when the system is empty, Borst [29] derives explicit expressions for the LST of the cycle-time distribution, for the LST of the waiting-time distribution at each queue and for the probability generating function (PGF) of the joint queue length at polling instants. He also shows that for this model the waiting time at each of the queues is smaller (in the increasing convex ordering sense) than in the ordinary non-dormant server case. In addition, Borst [29], [30] derives a PCL for dormant-server models with exhaustive, gated, 1-limited and globally-gated service, in which the server is allowed to rest at some of the queues. Based on this PCL, he compares the

waiting times in a symmetrical model in the dormant and the non-dormant server case.

In cases in which the server is allowed to stop only at some particular queues, one may consider the problem of determining a dormant set that optimizes the system performance with respect to some performance measure. Liu et al. [126] show that, in order to minimize the amount of unfinished work at any time, the server should serve each queue exhaustively. In addition, they show that in a fully symmetrical model the server should rest as soon as the system becomes empty (so that each queue is in the optimal dormant set). Borst [29], [30] gives a simple PCL-based heuristic for approximating the dormant set which minimizes the mean total amount of work in the system. In section 5.5 we will propose another simple and fast-to-evaluate approximation of the dormant set which minimizes an arbitrary weighted sum of the mean waiting times at the queues.
Eisenberg [79] makes a fundamental observation concerning the analysis of polling models with and without switch-over times and dormant-server polling models. He states that the fundamental issue in the analysis of polling models is *not* the presence or absence of switch-over times, but whether or not the server stops when the system is empty. The key observation is that (except for the case of strictly cyclic service) simply specifying the switch-over times as zero does not uniquely identify the model. As stated by Choudhury [61] for the case of a general service order table, the limiting behavior of the non-dormant server model as the switch-over times tend to zero depends on how this limit is approached. This limit depends on assumed limiting ratios of mean switch-over times between the different queues. It makes a difference to subsequent customers in what stage an arrival to an empty system is served. This limiting behavior can be uniquely identified by specifying particular rules for stopping and resuming service. Eisenbergs observation that the presence of switch-over times is not essential in the analysis of polling models is supported by results in a series of papers of Fuhrmann [87], Cooper et al. [72], Srinivasan et al. [155] and Borst and Boxma [33]. In these studies, simple relations between polling models with and without switch-over times are obtained for polling models that allow an MTBP-interpretation (cf. section 1.2.3). As a consequence, Eisenberg's observation challenges to analyze the performance of dormant-server polling models.

The goal of this chapter is to develop a means for analyzing the performance of dormant-server models and for quantifying the benefits from the opportunity of resting. In applications, this gain can then be compared with the loss due to an increased overhead involved in keeping track of queue lengths and of controlling the server. We will consider cyclic polling models in which the server is allowed to stop only at the queues which are in some dormant set. As soon as the system becomes empty, the server keeps on cycling until it reaches the next queue at which it is allowed to rest, provided the system is still empty at that time. As soon as the first customer arrives while the server is resting,

the server restarts moving along the cycle. In Eisenberg's nomenclature, the stopping rule can be indicated as 'Continue-*next*-home', and the starting rule as 'Resume-cycle'.
Evidently, in some situations the assumption that the server restarts moving along the cycle when a customer arrives in an empty system is rather unrealistic (e.g. in manufacturing and production environments). In those situations, it seems to make sense to move immediately to the queue at which the first customer has just arrived, provided this information is known to the server. However, we reemphasize that the main goal of this study is to investigate the benefits from the opportunity of resting in its own right. It is interesting to extend the results presented throughout this chapter to other restart rules.
Because for these models exact results are scarce (cf. [29], [95]), we will show how the model can be analyzeded with the PSA. The implementation of the present model into the PSA has revealed some interesting features for a further development of the PSA. The **0**-process (cf. section 2.3.3) is no longer irreducible, and generally possesses more than one recurrent class. It will be shown that this complication can be solved by changing the recursive order in which the coefficients of the power-series expansions of the state probabilities are computed.
Numerical experiments with the PSA have been performed to investigate the cost savings that can be made by the opportunity of resting. The experiments have revealed that the system performance can be strongly improved by allowing the server to rest at a queue, especially in lightly- and medium-loaded systems in which the switch-over times are significant.
In addition to the analysis of the performance of dormant-server models, we consider the problem of determining an optimal dormant set, where an arbitrary weighted sum of the mean waiting times at the queues has to be minimized. As the computation time needed to solve this problem grows rapidly in the number of queues, we propose to approximate the optimal dormant set by the light-traffic limit (when the arrival rates tend to zero). This heuristic approach is readily implementable and gives quite accurate results.

The remainder of this chapter is organized as follows. In section 5.2 the model is described in detail. In section 5.3 we show how the model can be analyzed by the PSA. In section 5.4 an overview of the numerical results is given. In section 5.5 we consider the problem of determining an optimal dormant set. We propose and test a simple and fast-to-evaluate approximation method to solve this optimization problem.

5.2 Model description

Consider the basic polling model discussed in section 1.3 with s infinite buffer queues, $Q_1, \ldots, Q_s$, and independent Poisson arrival processes with rates $\lambda_i = a_i\rho$, $i = 1, \ldots, s$. The service times of the customers at Q_i are assumed to have a Coxian distribution with parameter Ψ_i^1, $\pi_i^{1,\xi}$, $\mu_i^{1,\xi}$, $\xi = 1, \ldots, \Psi_i^1$. The

times needed by the server to switch from Q_{i-1} to Q_i are Coxian distributed with parameters Ψ_i^0, $\pi_i^{0,\xi}$, $\mu_i^{0,\xi}$, $\xi = 1, \ldots, \Psi_i^0$, $i = 1, \ldots, s$. Denote by $\sigma_i^{(k)}$ the k-th moment of the switch-over time needed by the server to move from Q_{i-1} to Q_i, $i = 1, \ldots, s$, $k = 1, 2$. Define by σ_k the k-th moment over the total switch-over time per cycle of the server along the queues, $k = 1, 2$. The server is allowed to rest at a queue when the entire system is empty. More specifically, we allow the server to rest at *some* of the queues; the set of indices corresponding to these queues, the dormant set, is denoted by $\mathcal{D}$. The number of customers that is served during a visit of the server to a queue is determined by a Bernoulli schedule $\boldsymbol{q} = (q_1, \ldots, q_s)$. When the server arrives at Q_h while it is empty or when the server has just emptied Q_h, the behavior of the server depends on whether the entire system is empty at that instant. If the system is non-empty at that particular instant, or if $h \notin \mathcal{D}$, then the server proceeds to the next queue; otherwise, the server rests at Q_h and waits until the next customer arrives. As soon as a customer arrives at the system, the server immediately resumes activities. If the first arriving customer arrives at Q_h, then that customer is taken into service immediately; otherwise, the server starts moving to the next queue.
Necessary and sufficient conditions for polling models in the non-dormant server case have been derived in [85]. Because in cases in which the ergodicity becomes critical the opportunity of resting arises with a probability which tends to 0, we suspect that the same conditions are necessary and sufficient in the dormant server case. For the present model these conditions read (cf. [85]):

$$\rho\,[1 + \sigma_1 a_i (1 - q_i)] < 1, \quad i = 1, \ldots, s. \tag{5.1}$$

In the sequel we will assume that condition (5.1) is satisfied and that the system is in the steady state.

Throughout this chapter we will need the notion of the following index sets. For a given $\mathcal{D} \neq \emptyset$ we partition the set $\{1, \ldots, s\}$ into the subsets $\{U_{\mathcal{D}}(i)\}_{i \in \mathcal{D}}$, defined by

$$U_{\mathcal{D}}(i) := \{i\} \cup \{1 \leq h \leq s \mid j \notin \mathcal{D}\ (j = h, \ldots, i-1)\} \quad (i \in \mathcal{D}). \tag{5.2}$$

In words, for $i \in \mathcal{D}$, $U_{\mathcal{D}}(i)$ corresponds to the set of queues h for which Q_i is the first queue in $\mathcal{D}$ that is visited after the server has started to move from Q_{h-1} to Q_h.

5.3 The power-series algorithm

In this section we show how the performance of the present model can be analyzed by means of the PSA. In 5.3.1 we define the state probabilities and formulate the global balance equations. In 5.3.2 the state probabilities are expressed as power series and a complete computational scheme to calculate the coefficients of these power series is derived.

5.3.1 Balance equations

For the case of a non-dormant server, the global balance equations for the present model are given in [22] (cf. also section 3.3.1). To extend these balance equations towards the dormant-server case, we adopt the same notation as in section 3.3.1. Denote by $\boldsymbol{N}(t) = (N_1(t), \ldots, N_s(t))$ the joint queue-length vector at time t, $t \geq 0$. In addition, define the vector of the supplementary variables $(H(t), Z(t), \Xi(t))$, where $H(t)$ denotes the index of the queue which is being served or is being switched to or at which the server is dormant at time t; the variable $Z(t)$ will indicate whether the server is dormant ($Z(t) = -1$) or switching ($Z(t)$=0) or serving ($Z(t)$=1) at time t; the variable $\Xi(t)$, which is only defined in the cases $Z(t)$=0 and $Z(t)$=1, indicates the actual phase number of either the switch-over time or the service time at time t, $t \geq 0$. We assume that the supplementary space is the same for all $\boldsymbol{n} \in \mathbb{N}^s$ and is given by

$$\begin{aligned} \mathcal{S} \;:=\; & \left\{(h,\zeta,\xi) \mid h = 1,\ldots,s,\;\; \zeta = 0,1,\;\; \xi = 1,\ldots,\Psi_h^{\zeta}\right\} \\ \cup\; & \{(h,-1) \mid h \in \mathcal{D}\}, \end{aligned} \tag{5.3}$$

while it is possible that some states can not be entered (cf. (5.10) below). Let $(\boldsymbol{N}, H, Z, \Xi)$ be a vector of random variables with as distribution the joint stationary distribution of $(\boldsymbol{N}(t), H(t), Z(t), \Xi(t))$.

We define the state probabilities as follows: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1,\ldots,s$, ζ=0,1, $\xi = 1,\ldots,\Psi_h^{\zeta}$,

$$p(\boldsymbol{n},h,\zeta,\xi) := \Pr\{(\boldsymbol{N},H,Z,\Xi) = (\boldsymbol{n},h,\zeta,\xi)\}; \tag{5.4}$$

and for $h \in \mathcal{D}$,

$$p(\boldsymbol{0},h,-1) := \Pr\{(\boldsymbol{N},H,Z) = (\boldsymbol{0},h,-1)\}, \tag{5.5}$$

i.e. the probability that the server is dormant at Q_h ($h \in \mathcal{D}$).

Equating the rate out of a state to the total rate into that state leads to the following set of balance equations for the state probabilities: for $\boldsymbol{n} \in \mathbb{N}^s$, $\boldsymbol{n} \neq \boldsymbol{0}$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$,

$$\begin{aligned} &\left[\rho \sum_{j=1}^{s} a_j + \mu_h^{0,\xi}\right] p(\boldsymbol{n},h,0,\xi) = \mu_h^{0,\xi+1} p(\boldsymbol{n},h,0,\xi+1) I\{\xi < \Psi_h^0\} \\ &+ \rho \sum_{j=1}^{s} a_j p(\boldsymbol{n}-\boldsymbol{e}_j,h,0,\xi) I\{n_j > 0\} \\ &+ \mu_{h-1}^{0,1} \pi_h^{0,\xi} p(\boldsymbol{n},h-1,0,1) I\{n_{h-1} = 0\} \\ &+ \mu_{h-1}^{1,1} \pi_h^{0,\xi} p(\boldsymbol{n}+\boldsymbol{e}_{h-1},h-1,1,1)\,[1 - q_{h-1} I\{n_{h-1} > 0\}] \\ &+ \rho \pi_h^{0,\xi} \sum_{j \neq h-1} a_j p(\boldsymbol{0},h-1,-1) I\{\boldsymbol{n} = \boldsymbol{e}_j\}\, I\{h-1 \in \mathcal{D}\}. \end{aligned} \tag{5.6}$$

The first term at the right-hand side of (5.6) indicates a phase transition of the switch-over time from Q_{h-1} to Q_h. The second term corresponds to a customer arrival while the server is switching from Q_{h-1} to Q_h. The third term describes an arrival of the server at Q_h, which is empty at that particular instant, so that the server directly proceeds to Q_h. The fourth term indicates a service completion at Q_{h-1}. The fifth term indicates an arrival at Q_j while the system is empty and the server is dormant at Q_{h-1}, $j \neq h-1$, so that the server immediately starts to move from Q_{h-1} to Q_h.

Similarly, for the states in which the server is switching while the system is empty, we have: for $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^0$,

$$\begin{aligned} &\left[\rho \sum_{j=1}^{s} a_j + \mu_h^{0,\xi}\right] p(\mathbf{0}, h, 0, \xi) = \mu_h^{0,\xi+1} p(\mathbf{0}, h, 0, \xi+1) I\left\{\xi < \Psi_h^0\right\} \\ &+ \left[\mu_{h-1}^{0,1} \pi_h^{0,\xi} p(\mathbf{0}, h-1, 0, 1) + \mu_{h-1}^{1,1} \pi_h^{0,\xi} p(\boldsymbol{e}_{h-1}, h-1, 1, 1)\right] \\ &\qquad \times I\left\{h-1 \notin \mathcal{D}\right\}. \end{aligned} \tag{5.7}$$

The first term at the right-hand side of (5.7) indicates a phase transition during a switch-over time from Q_{h-1} to Q_h while the system is empty. The second term describes that although the system is empty, the server skips Q_{h-1} because $h-1$ is not in the dormant set, and the server immediately starts to move to Q_h. The third term indicates that after the service of the only customer in the system at Q_{h-1}, the server proceeds to Q_h because the server is not allowed to rest at Q_{h-1}.

For the states in which the server is serving, we have: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^1$, $n_h > 0$,

$$\begin{aligned} &\left[\rho \sum_{j=1}^{s} a_j + \mu_h^{1,\xi}\right] p(\boldsymbol{n}, h, 1, \xi) = \mu_h^{1,\xi+1} p(\boldsymbol{n}, h, 1, \xi+1) I\left\{\xi < \Psi_h^1\right\} \\ &+ \rho \sum_{j=1}^{s} a_j p(\boldsymbol{n} - \boldsymbol{e}_j, h, 1, \xi) I\left\{n_j > 0\right\} + \mu_h^{0,1} \pi_h^{1,\xi} p(\boldsymbol{n}, h, 0, 1) \\ &+ q_h \mu_h^{1,1} \pi_h^{1,\xi} p(\boldsymbol{n} + \boldsymbol{e}_h, h, 1, 1) \\ &+ \rho a_h \pi_h^{1,\xi} p(\mathbf{0}, h, -1) I\left\{\boldsymbol{n} = \boldsymbol{e}_h\right\} I\left\{h \in \mathcal{D}\right\}. \end{aligned} \tag{5.8}$$

The first term at the right-hand side indicates a phase transition in a service of a customer at Q_h. The second term corresponds to an arrival during the service at Q_h. The third term indicates that the server arrives at Q_h and immediately starts serving at that queue. The fourth term indicates that after a service completion of a customer at Q_h the server immediately starts to serve the next customer at that queue. The fifth term marks an arrival at Q_h while the server is resting at that queue and an immediate service initiation at Q_h.

Finally, for the states in which the server is resting at a queue we have: for $h \in \mathcal{D}$,

$$\rho \sum_{j=1}^{s} a_j p(\mathbf{0}, h, -1) = \mu_h^{1,1} p(\boldsymbol{e}_h, h, 1, 1) + \mu_h^{0,1} p(\mathbf{0}, h, 0, 1). \tag{5.9}$$

Because the server can not be serving at an empty queue we have (cf. also (3.7)): for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^0$,

$$p(\boldsymbol{n}, h, 1, \xi) = 0, \text{ if } n_h = 0, \tag{5.10}$$

and according to the law of total probability, we have

$$\sum_{h \in \mathcal{D}} p(\mathbf{0}, h, -1) + \sum_{\boldsymbol{n} \in \mathbb{N}^s} \sum_{h=1}^{s} \sum_{\zeta=0}^{1} \sum_{\xi=1}^{\Psi_h^\zeta} p(\boldsymbol{n}, h, \zeta, \xi) = 1. \tag{5.11}$$

5.3.2 Computational scheme

In this section we derive a computational scheme for the PSA for the present model. The derivation proceeds along similar lines as discussed in section 2.3.3. We first express the state probabilities (5.4), (5.5) as power series. Then, these power-series expansions are substituted into the balance equations (5.6)-(5.9), leading to a set of linear relations between the coefficients of the power series. Finally, we define an ordering to calculate these coefficients recursively.

In section 2.3.2 necessary conditions for the applicability of the PSA have been given. These conditions only restrict the behavior of the system for non-empty states, so that it is irrelevant whether the server is allowed to rest at a queue when the system is empty. Hence, for the present model these *necessary* conditions are satisfied, so that the following light-traffic properties are valid for the present model: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, ζ=0,1, $\xi = 1, \ldots, \Psi_h^\zeta$,

$$p(\boldsymbol{n}, h, \zeta, \xi) = O(\rho^{|\boldsymbol{n}|}), \quad \rho \downarrow 0. \tag{5.12}$$

Moreover, we have for $h \in \mathcal{D}$,

$$p(\mathbf{0}, h, -1) = O(1), \quad \rho \downarrow 0. \tag{5.13}$$

However, these conditions are generally *not sufficient* for the applicability of the PSA. In this section we will show that the present model can still be analyzed by means of the PSA.

Based on properties (5.12) and (5.13), we introduce the following power-series expansions for the state probabilities: for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\zeta = 0, 1$, $\xi = 1, \ldots, \Psi_h^\zeta$,

$$p(\boldsymbol{n}, h, \zeta, \xi) = \rho^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \rho^k b_0(k; \boldsymbol{n}, h, \zeta, \xi), \tag{5.14}$$

and for $h \in \mathcal{D}$,

$$p(\mathbf{0}, h, -1) = \sum_{k=0}^{\infty} \rho^k b_0(k; \mathbf{0}, h, -1). \tag{5.15}$$

Substituting (5.14) and (5.15) into the balance equations (5.6), (5.7), (5.8) and (5.9), and equating corresponding powers of ρ, yields a set of linear relations between the coefficients of the power series. For later reference, for ζ=0 we explicitly distinguish the cases $\boldsymbol{n} = \mathbf{0}$, $\boldsymbol{n} = \boldsymbol{e}_i$ $(i = 1, \ldots, s)$ (cf. (5.17) and (5.18) below). The relations read as follows: for $\boldsymbol{n} \in \mathbb{N}^s$, $\boldsymbol{n} \neq \mathbf{0}$, $\boldsymbol{n} \neq \boldsymbol{e}_i$ $(i = 1, \ldots, s)$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^0$, $k = 0, 1, \ldots$,

$$\begin{aligned}
&\mu_h^{0,\xi} b_0(k; \boldsymbol{n}, h, 0, \xi) = \mu_h^{0,\xi+1} b_0(k; \boldsymbol{n}, h, 0, \xi + 1) I\{\xi < \Psi_h^0\} \\
&+ \sum_{j=1}^{s} a_j b_0(k; \boldsymbol{n} - \boldsymbol{e}_i, h, 0, \xi) I\{n_j > 0\} \\
&- \sum_{j=1}^{s} a_j b_0(k-1; \boldsymbol{n}, h, 0, \xi) I\{k > 0\} \\
&+ \mu_{h-1}^{0,1} \pi_h^{0,\xi} b_0(k; \boldsymbol{n}, h-1, 0, 1) I\{n_{h-1} = 0\} \\
&+ \mu_{h-1}^{1,1} \pi_h^{0,\xi} b_0(k-1; \boldsymbol{n} + \boldsymbol{e}_{h-1}, h-1, 1, 1) \\
&\qquad \times [1 - q_{h-1} I\{n_{h-1} > 0\}]\} I\{k > 0\};
\end{aligned} \tag{5.16}$$

for $i = 1, \ldots, s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^0$, $k = 0, 1, \ldots$,

$$\begin{aligned}
&\mu_h^{0,\xi} b_0(k; \boldsymbol{e}_i, h, 0, \xi) = \mu_h^{0,\xi+1} b_0(k; \boldsymbol{e}_i, h, 0, \xi + 1) I\{\xi < \Psi_h^0\} \\
&+ a_i b_0(k; \mathbf{0}, h, 0, \xi) - \sum_{j=1}^{s} a_j b_0(k-1; \boldsymbol{e}_j, h, 0, \xi) I\{k > 0\} \\
&+ \mu_{h-1}^{1,1} \pi_h^{0,\xi} b_0(k-1; \boldsymbol{e}_i + \boldsymbol{e}_{h-1}, h-1, 1, 1) \\
&\qquad \times [1 - q_{h-1} I\{h - 1 = i\}] I\{k > 0\} \\
&+ \Big[\mu_{h-1}^{0,1} \pi_h^{0,\xi} b_0(k; \boldsymbol{e}_i, h-1, 0, 1) \\
&\qquad + \pi_h^{0,\xi} a_i b_0(k; \mathbf{0}, h-1, 0, -1) I\{h - 1 \in \mathcal{D}\}\Big] I\{h - 1 \neq i\};
\end{aligned} \tag{5.17}$$

for $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^0$, $k = 0, 1, \ldots$,

$$\begin{aligned}
&\mu_h^{0,\xi} b_0(k; \mathbf{0}, h, 0, \xi) = \mu_h^{0,\xi+1} b_0(k; \mathbf{0}, h, 0, \xi + 1) I\{\xi < \Psi_h^0\} \\
&- \sum_{j=1}^{s} a_j b_0(k-1; \mathbf{0}, h, 0, \xi) I\{k > 0\} \\
&+ \Big[\mu_{h-1}^{0,1} \pi_h^{0,\xi} b_0(k; \mathbf{0}, h-1, 0, 1) \\
&\quad + \mu_{h-1}^{1,1} \pi_h^{0,\xi} b_0(k-1; \boldsymbol{e}_{h-1}, h-1, 1, 1) I\{k > 0\}\Big] I\{h - 1 \notin \mathcal{D}\};
\end{aligned} \tag{5.18}$$

for $\boldsymbol{n} \in \mathbb{N}^s$, $h = 1, \ldots, s$, $\xi = 1, \ldots, \Psi_h^1$, $n_h > 0$, $k = 0, 1, \ldots$,

$$
\begin{aligned}
&\mu_h^{1,\xi} b_0(k;\boldsymbol{n},h,1,\xi) = \mu_h^{1,\xi+1} b_0(k;\boldsymbol{n},h,1,\xi+1) I\{\xi < \Psi_h^1\} \\
&+ \sum_{j=1}^{s} a_j b_0(k;\boldsymbol{n}-\boldsymbol{e}_j,h,1,\xi) I\{n_j > 0\} \\
&- \sum_{j=1}^{s} a_j b_0(k-1;\boldsymbol{n},h,1,\xi) I\{k > 0\} \\
&+ \mu_h^{0,1} \pi_h^{1,\xi} b_0(k;\boldsymbol{n},h,0,1) \\
&+ q_h \mu_h^{1,1} \pi_h^{1,\xi} b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_h,h,1,1) I\{k > 0\} \\
&+ a_h \pi_h^{1,\xi} b_0(k;\boldsymbol{0},h,-1) I\{\boldsymbol{n} = \boldsymbol{e}_h\} I\{h \in \mathcal{D}\};
\end{aligned}
\tag{5.19}
$$

and for $h \in \mathcal{D}$, $k = 0,1,\ldots,$

$$
\begin{aligned}
\sum_{j=1}^{s} a_j b_0(k;\boldsymbol{0},h,-1) &= \mu_h^{1,1} b_0(k;\boldsymbol{e}_h,h,1,1) \\
&+ \mu_h^{0,1} b_0(k+1;\boldsymbol{0},h,0,1).
\end{aligned}
\tag{5.20}
$$

For $\boldsymbol{n} \neq \boldsymbol{0}$ and $\boldsymbol{n} \neq \boldsymbol{e}_i$ $(i = 1,\ldots,s)$, relations (5.16) and (5.19) express the coefficients $b_0(k;\boldsymbol{n},h,\zeta,\xi)$ in terms of lower order with respect to the following partial ordering $\prec$ of the vectors $(k;\boldsymbol{n},h,\zeta,\xi)$: for $(k;\boldsymbol{n},h,\zeta,\xi)$, $(\hat{k},\hat{\boldsymbol{n}},\hat{h},\hat{\zeta},\hat{\xi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ (cf. also (2.12)),

$$
\begin{aligned}
&(k;\boldsymbol{n},h,\zeta,\xi) \prec (\hat{k},\hat{\boldsymbol{n}},\hat{h},\hat{\zeta},\hat{\xi}) \\
&\text{if } \left[k+|\boldsymbol{n}| < \hat{k}+|\hat{\boldsymbol{n}}|\right] \vee \left[k+|\boldsymbol{n}| = \hat{k}+|\hat{\boldsymbol{n}}| \wedge k < \hat{k}\right].
\end{aligned}
\tag{5.21}
$$

We will now define an ordering for the supplementary variables $(h,\zeta,\xi) \in \mathcal{S}$. For given $(k;\boldsymbol{n}) \in \mathbb{N}^{1+s}$ and $h = 1,\ldots,s$, define (cf. also (3.15)):

$$
(k;\boldsymbol{n},h,\zeta,\xi) \prec (k;\boldsymbol{n},h,\hat{\zeta},\hat{\xi}) \text{ if } \left[\zeta < \hat{\zeta}\right] \vee \left[\zeta = \hat{\zeta} \wedge \xi > \hat{\xi}\right]. \tag{5.22}
$$

To define an ordering of the variables $h = 1,\ldots,s$, note that because $\boldsymbol{n} \neq \boldsymbol{0}$, there exists some index i such that $n_{i-1} > 0$. Let $h^*_{\boldsymbol{n}} := i$. Then we define the ordering of the vectors $(k;\boldsymbol{n},h,\zeta,\xi)$ over the components $h = 1,\ldots,s$ as follows: for $(k;\boldsymbol{n},h,\zeta,\xi)$, $(k;\boldsymbol{n},\hat{h},\hat{\zeta},\hat{\xi}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ (cf. also (3.17)),

$$
\begin{aligned}
&(k;\boldsymbol{n},h,\zeta,\xi) \prec (k;\boldsymbol{n},\hat{h},\hat{\zeta},\hat{\xi}) \\
&\text{if } [\boldsymbol{n} \neq \boldsymbol{0}] \wedge \left[(h-h^*_{\boldsymbol{n}}) \bmod s \;<\; (\hat{h}-h^*_{\boldsymbol{n}}) \bmod s\right].
\end{aligned}
\tag{5.23}
$$

Hence, it remains to extend the ordering to the states with $\zeta = 0$ and either $\boldsymbol{n} = \boldsymbol{0}$ or $\boldsymbol{n} = \boldsymbol{e}_i$ $(i = 1,\ldots,s)$, the states with $\zeta = -1$, and the states with $\zeta = 1$ and $\boldsymbol{n} = \boldsymbol{e}_i$ $(i = 1,\ldots,s)$.

Let us first consider the states with $\zeta = 0$ and $\boldsymbol{n} = \boldsymbol{0}$ and let k be fixed, $k = 0,1,\ldots$. It follows from (5.8) that if $\mathcal{D} \neq \emptyset$, then according to the ordering $\prec$ in (5.21)-(5.23), the coefficients $b_0(k;\boldsymbol{0},h,0,\xi)$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$, can be calculated recursively, starting with a coefficient $b_0(k;\boldsymbol{0},h,0,\Psi_h^0)$ for which $h-1 \in \mathcal{D}$. If $\mathcal{D} = \emptyset$ then a set of s linear equations has to be solved to

compute the coefficients $b_0(k;\mathbf{0},h,0,\xi)$, $h=1,\ldots,s$, $\xi=1,\ldots,\Psi_h^0$ (cf. also the derivation in section 3.3.2).

As for the coefficients $b_0(k;\mathbf{0},h,-1)$, $h\in\mathcal{D}$, $k=0,1,\ldots$, substituting (5.20) into (5.18) and (5.19) leads to the following set of linear equations for the coefficients $b_0(k;\mathbf{0},h,-1)$, $h\in\mathcal{D}$:

$$\sum_{i=1}^{s} a_i b_0(k;\mathbf{0},h,-1) = \sum_{j\in\mathcal{D}} \sum_{i\in U_{\mathcal{D}}(h)} a_i b_0(k;\mathbf{0},j,-1) + y_0(k;h), \tag{5.24}$$

where $y_0(0;h):=0$, $h\in\mathcal{D}$, and for $h\in\mathcal{D}$, $k=1,2,\ldots$,

$$\begin{aligned} y_0(k;h) :=& -\left(\sum_{j=1}^{s} a_j\right) \\ &\times \sum_{i\in U_{\mathcal{D}}(h)} \left\{ \sum_{\xi=1}^{\Psi_i^1} b_0(k-1;\boldsymbol{e}_i,i,1,\xi) + \sum_{j=1}^{s}\sum_{\xi=1}^{\Psi_j^0} b_0(k-1;\boldsymbol{e}_i,j,0,\xi) \right\} \\ &+ \sum_{i\in U_{\mathcal{D}}(h)} \sum_{j=1}^{s} \mu_j^{1,1} b_0(k-1;\boldsymbol{e}_i+\boldsymbol{e}_j,j,1,1) \\ &+ \sum_{i\in U_{\mathcal{D}}(h)} a_i \sum_{j=1}^{s}\sum_{\xi=1}^{\Psi_j^0} b_0(k;\mathbf{0},j,0,\xi) \\ &- \sum_{j=1}^{s} a_j \sum_{i\in U_{\mathcal{D}}(h)} \sum_{\xi=1}^{\Psi_i^0} b_0(k;\mathbf{0},i,0,\xi). \end{aligned} \tag{5.25}$$

However, one may verify, by summing over $h\in\mathcal{D}$, that the set of equations (5.24) is dependent. An additional linear equation follows from the law of total probability (5.11). Substituting (5.14) and (5.15) into (5.11) implies:

$$\sum_{h\in\mathcal{D}} b_0(k;\mathbf{0},h,-1) = Y_0(k), \tag{5.26}$$

where $Y_0(0):=1$ and for $k=1,2,\ldots$,

$$\begin{aligned} Y_0(k) :=& -\sum_{0<|\boldsymbol{n}|\le k} \sum_{h=1}^{s}\sum_{\zeta=0}^{1}\sum_{\xi=1}^{\Psi_h^\zeta} b_0(k-|\boldsymbol{n}|;\boldsymbol{n},h,\zeta,\xi) \\ &- \sum_{h=1}^{s}\sum_{\xi=1}^{\Psi_h^0} b_0(k;\mathbf{0},h,0,\xi). \end{aligned} \tag{5.27}$$

To verify the validity of the set of equations (5.26) for k=0, it should be noted that it follows directly from (5.18) that $b_0(0;\mathbf{0},h,0,\xi)=0$ for $h=1,\ldots,s$, $\xi=1,\ldots,\Psi_h^0$. To obtain the coefficients $b_0(k;\mathbf{0},h,-1)$, $h\in\mathcal{D}$, from the set of equations (5.24) and (5.26), the coefficients in $y_0(k;h)$ have to be known. That is, for each $h\in\mathcal{D}$, $k=0,1,\ldots$, the coefficients $b_0(k;\mathbf{0},h,-1)$ have to be of higher order than all coefficients in (5.25). To this end, we extend the ordering $\prec$, defined in (5.21)-(5.23), to the states in which the server is resting as follows: for $(k;\boldsymbol{n},h,\zeta,\xi),(\hat{k},\mathbf{0},h,-1)\in\mathbb{N}^{1+s}\times\mathcal{S}$,

$$(k;\boldsymbol{n},h,\zeta,\xi) \prec (\hat{k},\mathbf{0},h,-1) \text{ if } \hat{k} \geq k+|\boldsymbol{n}|-1. \tag{5.28}$$

Once the coefficients $b_0(k;\mathbf{0},h,-1)$, $h \in \mathcal{D}$, have been determined, the coefficients $b_0(k;\boldsymbol{e}_l,h,0,\xi)$, $l,h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$, and the coefficients $b_0(k;\boldsymbol{e}_h,h,1,\xi)$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^1$, can be computed recursively according to (5.17) and (5.19) respectively, $k = 0,1,\ldots$.

Let $g^{(l)}(\boldsymbol{n},\boldsymbol{\varphi})$ be an arbitrary real-valued function of the state space. Then general performance measures of the form $\mathrm{E}\left\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\right\}$, $l = 1,\ldots,L$, can be expressed in terms of the coefficients of the power series as (cf. also (2.18), (2.19)): for $l = 1,\ldots,L$,

$$\mathrm{E}\left\{g^{(l)}(\boldsymbol{N},\boldsymbol{\Phi})\right\} = \sum_{k=0}^{\infty} \rho^k f^{(l)}(k), \tag{5.29}$$

where for $k = 0,1,\ldots$,

$$\begin{aligned} f^{(l)}(k) := & \\ & \sum_{0\leq|\boldsymbol{n}|\leq k} \sum_{h=1}^{s} \sum_{\zeta=0}^{1} \sum_{\xi=1}^{\Psi_h^\zeta} g^{(l)}(\boldsymbol{n},h,\zeta,\xi) b_0(k-|\boldsymbol{n}|;\boldsymbol{n},h,\zeta,\xi) \\ & + \sum_{h\in\mathcal{D}} g^{(l)}(\mathbf{0},h,-1) b_0(k;\mathbf{0},h,-1). \end{aligned} \tag{5.30}$$

To compute these performance measures up to the, say, M-th power of ρ, the following computational scheme should be applied.

step 1 : $m := 0$; $f^{(l)}(k) := 0$, $l = 1,\ldots,L$, $k = 0,1,\ldots,M$;

step 2 : determine $b_0(m;\mathbf{0},h,0,\xi)$, $h = 1,\ldots,s$, $\xi = 1,\ldots,\Psi_h^0$, according to (5.18), and update $f^{(l)}(m)$, $l = 1,\ldots,L$, according to (5.30);

step 3 : for all $(k;\boldsymbol{n},h,\zeta,\xi)$ for which $k+|\boldsymbol{n}| = m+1$, $\boldsymbol{n} \neq \mathbf{0}$, $\boldsymbol{n} \neq \boldsymbol{e}_i$ $(i = 1,\ldots,s)$, determine $b_0(k;\boldsymbol{n},h,0,\xi)$, $h = 1,\ldots,s$, $\xi = \Psi_h^0,\ldots,1$, according to (5.16) and $b_0(k;\boldsymbol{n},h,1,\xi)$, $h = 1,\ldots,s$, $\xi = \Psi_h^1,\ldots,1$, according to (5.19), in increasing order of $(k;\boldsymbol{n},h,\zeta,\xi)$ with respect to $\prec$ (cf. (5.21)-(5.23)), and update $f^{(l)}(m+1)$, $l = 1,\ldots,L$, according to (5.30);

step 4 : determine $b_0(m;\mathbf{0},h,-1)$, $h \in \mathcal{D}$, according to (5.24) and (5.26), and update $f^{(l)}(m)$, $l = 1,\ldots,L$, according to (5.30);

step 5 : determine $b_0(m;\boldsymbol{e}_l,h,0,\xi)$, $h,l = 1,\ldots,s$, $\xi = \Psi_h^0,\ldots,1$, according to (5.17), and update $f^{(l)}(m+1)$, $l = 1,\ldots,L$, according to (5.30);

step 6 : determine $b_0(m;\boldsymbol{e}_h,h,1,\xi)$, $h = 1,\ldots,s$, $\xi = \Psi_h^1,\ldots,1$, according to (5.19), and update $f^{(l)}(m+1)$, $l = 1,\ldots,L$, according to (5.30);

step 7 : $m := m + 1$; if $m < M$ then return to *step 2*; otherwise, STOP.

For $\mathcal{D} \neq \emptyset$, the global balance equations (5.6)-(5.9), and hence the linear relations (5.16)-(5.19), differ slightly from those in the case $\mathcal{D} = \emptyset$, cf. (3.5)-(3.6) and (3.12)-(3.13), respectively. For the case $\mathcal{D} = \emptyset$, *step 4* in the computational scheme vanishes, and it is readily verified that the resulting computational scheme corresponds exactly to the one discussed in section 3.3.

The computational scheme discussed here can readily be extended to the computation of derivatives with respect to continuous system parameters, as elaborated upon in chapter 2.

Let us consider the light-traffic behavior of the system. It follows from (5.24), (5.26), $Y_0(0)$=1 and $y_0(0;h) = 0$ $(h \in \mathcal{D})$ that for $h \in \mathcal{D}$,

$$b_0(0;\mathbf{0},h,-1) = \frac{\sum\limits_{i \in U_{\mathcal{D}}(h)} a_i}{\sum\limits_{i=1}^{s} a_i} = \frac{\sum\limits_{i \in U_{\mathcal{D}}(h)} \lambda_i}{\sum\limits_{i=1}^{s} \lambda_i}. \tag{5.31}$$

From the power-series expansions for the state probabilities corresponding to the resting states (5.5), it follows that for $h \in \mathcal{D}$,

$$b_0(0;\mathbf{0},h,-1) = \lim_{\rho \downarrow 0} p(\mathbf{0},h,-1), \tag{5.32}$$

are the light-traffic limits of the dormant server states (where the ratios between the arrival rates are kept fixed). Expression (5.31) follows from (5.32) by noting that $U_{\mathcal{D}}(h)$ is the set of queues i for which Q_h is the first queue at which the server is allowed to rest after it has started to move from Q_{i-1} to Q_i.

In the non-dormant server case the coefficients with $(k;\boldsymbol{n}) = (0;\mathbf{0})$ can directly be obtained by considering the process for $\rho = 0$, the **0**-process. In that case, the coefficients $b_0(0;\mathbf{0},h,\zeta,\xi)$ simply correspond to the state probabilities in the **0**-process. A *sufficient* condition for the existence of these state probabilities is that the **0**-process is *irreducible* on the subset of reachable states. Clearly, in the non-dormant server case this condition is satisfied. However, in the dormant server case the **0**-process is *not* irreducible and has generally more than one recurrent class (each uniquely corresponding to some $h \in \mathcal{D}$), so that the **0**-process does not possess a unique stationary distribution, unless $|\mathcal{D}| = 1$. Thus, for dormant server models it is *not* sufficient to consider the **0**-process to determine the light-traffic limits of the state probabilities according to (5.32), except when $\mathcal{D}$ is a singleton. Hence, also coefficients corresponding to non-empty states have to be taken into account to determine the coefficients $b_0(0;\mathbf{0},h,-1)$, $h \in \mathcal{D}$ (cf. (5.32)), so that the order in which the coefficients are computed has to be modified.
This need for modification of the ordering of the coefficients becomes apparent in (5.20), where the coefficients $b_0(k;\mathbf{0},h,-1)$ are not expressed in terms of

lower order with respect to the partial ordering over the couples $(k; \boldsymbol{n})$ in (2.12). To modify the ordering in such a way that (5.20) *does* express the coefficients $b_0(k; \mathbf{0}, h, -1)$ in terms of lower order, the ranking of the states $(k; \mathbf{0}, h, -1)$ has to be *upgraded* such that these states have the highest ranking among all states with $(\hat{k}, \boldsymbol{n})$ with $\hat{k} + |\boldsymbol{n}| = k + 1$ (cf. (5.28)). This modification of the computation order ensures that all terms at the right-hand side of (5.25) are of lower order than the terms $b_0(k; \mathbf{0}, h, -1)$, so that these terms can be considered to be known in (5.25).

Let us consider the solvability of the set of equations (5.24), (5.26) with unknown variables $b_0(k; \mathbf{0}, h, -1)$, $h \in \mathcal{D}$. Because the solvability of this set of equations does not depend on k, suppose k=0 and consider the *discrete-time* Markov process D on the state space $\mathcal{D}$ and with the following transition probabilities: for $h_1, h_2 \in \mathcal{D}$,

$$p_{h_1,h_2} = \frac{\sum\limits_{i \in U_{\mathcal{D}}(h_2)} a_i}{\sum\limits_{i=1}^{s} a_i}, \tag{5.33}$$

independent of h_1. From $y_0(0; h) = 0$ and $Y_0(0) = 1$, it follows that the set of equations (5.24) and (5.26) possesses a unique solution if and only if the Markov process D possesses a unique stationary distribution (cf. e.g. [148]). Clearly, the latter condition is satisfied, because generally $a_i > 0$, $i = 1, \ldots, s$. Note that if $|\mathcal{D}|$=1 the left-hand side of (5.24) is expressed in terms of lower order only, so that the coefficients can be directly obtained without having to solve a set of linear equations.

5.4 Numerical results

In this section the extension of the PSA, discussed in section 5.3, is used to investigate the performance of models in which the server is allowed to rest at a queue when the system is empty. We will give some rough guidelines on the qualitative behavior of the system performance, illustrated by numerical examples. Throughout, we take as performance measure the following cost function:

$$C(\mathcal{D}) := \sum_{i=1}^{s} c_i \mathrm{E}W_i, \tag{5.34}$$

where $\boldsymbol{c} = (c_1, \ldots, c_s)$ is an arbitrary vector of non-negative weights. Clearly, the mean waiting times generally depend on the choice of the dormant set $\mathcal{D}$.

In the numerical examples discussed in this section, we will take $c_i := \rho_i/\rho$, $i = 1, \ldots, s$, so that $C(\mathcal{D})$ is proportional to the mean amount of waiting work in the system. Unless indicated otherwise, q_i will be taken to be 1.00 for all i, i.e. all queues are served exhaustively, and the service times and switch-over

times will be assumed to be exponentially distributed. Denote the dormant set that minimizes $C(\mathcal{D})$ over all subsets of $\{1,\ldots,s\}$ by $\mathcal{D}^*$, so that $\mathcal{D}^*$ satisfies

$$C(\mathcal{D}^*) = \min_{\mathcal{D}\subset\{1,\ldots,s\}} C(\mathcal{D}). \tag{5.35}$$

Throughout this section, the symbol '%' indicates the relative improvement that can be made by allowing the server to rest, defined by

$$\frac{C(\emptyset) - C(\mathcal{D}^*)}{C(\emptyset)} \times 100. \tag{5.36}$$

Note that $C(\emptyset)$ is the cost function (5.34) in the non-dormant server case.
The numerical examples presented here are based on the use of the PSA. The number of terms of the power series that has been computed varies from $M = 15$ (for $\rho \leq 0.5$) to $M = 40$ (for $\rho \geq 0.8$). In all cases, the estimated absolute error in the computations is less than 0.001. In the numerical examples considered here, the optimal dormant set has been obtained by complete enumeration of the cost function for all possible dormant sets.

Offered load
To examine the influence of the offered load to the system on the improvements of the performance of the system that can be achieved by allowing the server to rest, $C(\emptyset)$ and $C(\mathcal{D}^*)$ have been computed for various values of the offered load for a model with the following set of parameters: $s = 3$; all arrival rates are equal; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/3, \sigma_1/3, \sigma_1/3)$; all switch-over times are Coxian distributed with squared coefficient of variation α. For all these models, we have $\mathcal{D}^* = \{1,2,3\}$ (cf. [126]). Tables 5.1 and 5.2 show the waiting cost (5.34) for various values of ρ and α.

	$\alpha = 1.00$			$\alpha = 5.00$			$\alpha = 20.00$		
ρ	$C(\emptyset)$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$C(\mathcal{D}^*)$	%
0.10	1.04	0.69	34.1	2.17	0.81	65.5	5.92	0.95	83.9
0.30	1.52	1.21	20.1	2.64	1.42	46.2	6.39	2.10	67.2
0.50	2.38	2.13	10.2	3.50	2.52	28.1	7.25	3.74	48.5
0.70	4.38	4.21	3.9	5.50	4.79	13.0	9.25	6.62	28.4
0.90	14.38	14.29	0.6	15.50	15.12	2.5	19.25	17.83	7.4
0.95	29.38	29.32	0.2	30.50	30.24	0.9	34.25	33.31	2.7

Table 5.1: Dormant and non-dormant server models; σ_1=1.50.

Tables 5.1 and 5.2 show that the performance of the system can be strongly improved by allowing the server to rest when the entire system is empty. Moreover, it is shown that the relative decrease of the waiting cost decreases when the offered load to the system is increased. The latter is due to the fact that the opportunity of resting occurs less frequently when the offered load is increased.

	$\alpha = 1.0$			$\alpha = 5.0$			$\alpha = 20.0$		
ρ	$C(\emptyset)$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$C(\mathcal{D}^*)$	%
0.10	3.83	2.64	31.1	8.33	3.55	57.5	23.33	6.45	72.4
0.30	4.79	4.17	13.0	9.39	6.55	29.5	24.29	14.05	42.2
0.50	6.50	6.25	3.8	11.00	9.62	12.5	26.00	20.29	22.0
0.70	10.50	10.43	0.6	15.00	14.45	3.7	30.00	27.42	8.6
0.90	30.50	30.50	0.0	35.00	34.92	0.3	50.00	48.61	2.8
0.95	60.50	60.50	0.0	65.00	64.99	0.1	80.00	77.93	2.6

Table 5.2: Dormant and non-dormant server models; σ_1=6.00.

Mean switch-over times

To examine the influence of the mean switch-over times on the relative improvements that can be made by resting, we have computed the performance of a model with the following set of parameters: $s = 3$; $\boldsymbol{a} = (0.50, 0.25, 0.25)$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$; $\boldsymbol{\sigma}^{(1)} = (\sigma_1/2, \sigma_1/4, \sigma_1/4)$, for various values of ρ and σ_1. Tables 5.3 and 5.4 show the performance measures for $\boldsymbol{q} = (0.00, 0.00, 0.00)$ and $\boldsymbol{q} = (1.00, 1.00, 1.00)$ respectively. Tables 5.3 and 5.4 illustrate that when the mean switch-over times are increased, starting with $\sigma_1 = 0.00$, the relative improvement of the performance increases for low values of σ_1 and decreases for higher values of σ_1. This phenomenon is due to the trade-off between the fact that possible improvements would increase for increasing values of σ_1 on the one hand and the fact that the opportunity of resting occurs less frequently for increasing values of σ_1 on the other hand.

	$\rho = 0.3$				$\rho = 0.7$			
σ_1	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%
0.20	0.64	$\{1\}$	0.55	15.3	3.47	$\{1\}$	3.37	2.7
1.00	1.69	$\{1\}$	1.22	27.7	∞	-	∞	-
2.00	3.65	$\{1\}$	2.79	23.6	∞	-	∞	-
4.00	19.92	$\{1\}$	18.63	6.5	∞	-	∞	-
6.00	∞	-	∞	-	∞	-	∞	-
8.00	∞	-	∞	-	∞	-	∞	-

Table 5.3: Influence of the mean switch-over times; 1-limited service.

Variability of the switch-over times

To examine the effect of the variability of the switch-over times, consider the same model as in Table 5.4, with $\boldsymbol{\sigma}^{(1)}$=(0.50,0.25,0.25). Moreover, we assume that the switch-over times to move from Q_2 to Q_3 and from Q_3 to Q_1 are

	$\rho = 0.3$				$\rho = 0.7$			
σ_1	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%
0.20	0.59	{1}	0.50	16.3	2.62	{1}	2.53	3.5
1.00	1.25	{1}	0.84	33.1	3.75	{1}	3.49	7.1
2.00	2.07	{1}	1.39	33.0	5.17	{1,2}	4.88	5.6
4.00	3.71	{1}	2.78	25.3	8.00	{1,2,3}	7.79	2.6
6.00	5.36	{1,2}	4.34	19.0	10.83	{1,2,3}	10.70	1.3
8.00	7.00	{1,2}	5.99	14.4	13.67	{1,2,3}	15.38	0.6

Table 5.4: Influence of the mean switch-over times; exhaustive service.

exponentially distributed and that the times needed by the server to move from Q_1 to Q_2 are 2-phase Coxian distributed with squared coefficient of variation α. Table 5.5 shows the values of the performance measures for various values of ρ and α.

	$\rho = 0.3$				$\rho = 0.7$			
α	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%
0.25	1.16	{1}	0.82	29.1	3.66	{1}	3.44	5.9
0.50	1.19	{1}	0.82	30.5	3.69	{1}	3.46	6.2
1.00	1.25	{1}	0.84	33.2	3.75	{1}	3.49	7.1
5.00	1.75	{1}	0.92	47.5	4.25	{1,2}	3.70	12.9
10.00	2.38	{1}	1.02	60.1	4.89	{1,2,3}	3.95	19.1
20.00	3.63	{1,2}	1.20	67.0	6.13	{1,2,3}	4.41	28.0

Table 5.5: Influence of the variability in the switch-over times.

Table 5.5 illustrates that the benefit of the opportunity of resting increases considerably with increasing variability of the switch-over times.

Asymmetry in the arrival rates

Finally, we examine the effect of the asymmetry in the arrival rates. To this end, consider the polling model with the following set of parameters: $s = 3$; $\boldsymbol{a} = (\alpha/(\alpha+2), 1/(\alpha+2), 1/(\alpha+2))$, so that the ratios between the arrival rates are α:1:1; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00)$; all service times are exponentially distributed; $\boldsymbol{\sigma}^{(1)} = (0.50, 0.25, 0.25)$; the switch-over times from Q_1 to Q_2 are 2-phase Coxian distributed with squared coefficient of variation 4.00; all other switch-over times are exponentially distributed. Table 5.6 shows the performance of the system for various values of ρ and α.

Table 5.6 illustrates that the benefit of the opportunity of resting increases with increasing asymmetry of the arrival process.

	$\rho = 0.3$				$\rho = 0.7$			
α	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%	$C(\emptyset)$	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	%
0.05	0.64	{2}	0.91	37.4	4.09	{2,3}	3.60	11.8
0.25	1.69	{1}	0.99	39.2	4.17	{1,2,3}	3.74	10.4
0.50	3.65	{1}	0.90	44.7	4.13	{1,2}	3.65	11.5
1.00	19.92	{1}	0.83	48.6	4.05	{1}	3.51	13.2
1.50	∞	{1}	0.73	53.8	3.92	{1}	3.29	16.1
2.00	∞	{1}	0.62	60.1	3.74	{1}	2.98	20.3

Table 5.6: Influence of the asymmetry in the arrival rates.

We emphasize that, in practice, the variability of the switch-over times will generally be smaller than in the examples considered in this section. However, we have constructed the examples such that the various effects are illustrated clearly. In general, the benefits of the possibility of resting decrease with decreasing variability of the switch-over times (cf. Table 5.5 in this section and [30]).

5.5 Optimization

In this section we consider the following optimization problem:

$$\min_{\mathcal{D} \subset \{1,\ldots,s\}} C(\mathcal{D}), \tag{5.37}$$

where $C(\mathcal{D})$ is defined in (5.34). In words, the problem is to determine a dormant set that minimizes an arbitrary weighted sum of the mean waiting times at the queues. The optimal dormant set will be denoted by $\mathcal{D}^*$.

This problem might be solved numerically either by formulating the problem as a semi-Markov decision problem (cf. remark 4.1 of [29]) or by complete enumeration of the performance measure for all 2^s subsets of $\mathcal{D}$. However, both approaches become rather time consuming when the number of queues in the system becomes large. For this reason, we propose and test in this section a simple approximation, for which the time requirements are negligible and which yields fairly accurate results over a wide range of admissible model parameters. This approximation is based on the observation that the choice of an appropriate dormant set $\mathcal{D}$ is most critical in light-traffic models. Motivated by this observation, we will propose to approximate the optimal dormant set $\mathcal{D}^*$ by its light-traffic limit, which is very easy to determine.

Denote by π_h the light-traffic limit of the probability that the server is dormant at Q_h, $h = 1, \ldots, s$; here, the limits are taken in such a way that the ratios between the arrival rates are kept fixed ($\lambda_i = a_i \rho$, $i = 1, \ldots, s$, where the value

of ρ is varied). Note that $\pi_h = b_0(0; \mathbf{0}, h, -1)$ for $h \in \mathcal{D}$ (cf. (5.31)) and $\pi_h = 0$ for $h \notin \mathcal{D}$. Then the light-traffic limit of the waiting-cost function (5.34) can be expressed as follows (by conditioning on the queue at which the server is resting): for $\mathcal{D} \neq \emptyset$,

$$\begin{aligned} \lim_{\rho \downarrow 0} C(\mathcal{D}) &= \sum_{i=1}^{s} c_i \sum_{h \in \mathcal{D} \setminus \{i\}} \pi_h \sum_{k=h+1}^{i} \sigma_k^{(1)} \\ &= \sum_{h \in \mathcal{D}} \pi_h \sum_{i \neq h} c_i \sum_{k=h+1}^{i} \sigma_k^{(1)} \\ &= \sum_{h \in \mathcal{D}} \pi_h \gamma_h, \end{aligned} \tag{5.38}$$

where for $h = 1, \ldots, s$,

$$\gamma_h := \sum_{i \neq h} c_i \sum_{k=h+1}^{i} \sigma_k^{(1)}. \tag{5.39}$$

Moreover, for $\mathcal{D} = \emptyset$ we have (cf. also (3.29))

$$\lim_{\rho \downarrow 0} C(\emptyset) = \frac{\sigma_2}{2\sigma_1}. \tag{5.40}$$

Using the definition of γ_h $(h = 1, \ldots, s)$ in (5.39), we define

$$\mathcal{T} := \{i \in \{1, \ldots, s\} \mid \gamma_i \leq \gamma_j \quad (j = 1, \ldots, s)\}, \tag{5.41}$$

i.e. the set of queues h for which γ_h is minimal over all queues. To verify that $\mathcal{T}$ indeed minimizes (5.38) and (5.40) over all subsets of $\{1, \ldots, s\}$, it should also be noted that

$$\sum_{h=1}^{s} \sum_{i \neq h} \sum_{k=h+1}^{i} c_i \sigma_k^{(1)} = \frac{1}{2} s \sigma_1 - \sum_{i=1}^{s} \sigma_i^{(1)} c_i, \tag{5.42}$$

so that

$$\min_{h=1,\ldots,s} \sum_{i \neq h} c_i \sum_{k=h+1}^{i} \sigma_k^{(1)} \leq \frac{1}{2}\sigma_1 \leq \frac{\sigma_2}{2\sigma_1}. \tag{5.43}$$

This inequality implies that (5.40) can not be strictly smaller than the minimal value of (5.38). Hence, in all cases there exists a non-empty dormant set for which the light-traffic limit of the cost function is smaller than for the empty dormant set.

Motivated by the above-mentioned observation that the choice of an appropriate dormant set is most critical in light-traffic models, and because the index set $\mathcal{T}$ (cf. (5.41)) is particularly easy to determine, we propose to approximate the optimal dormant set $\mathcal{D}^*$ by

$$\mathcal{D}^*(app) := \mathcal{T}. \tag{5.44}$$

This approximation is very easy to implement and proceeds as follows:

$min := \infty$; $\mathcal{T} := \emptyset$;
for $h := 1$ to s do
  begin
    determine γ_h;
    if $\gamma_h < min$ then $min := \gamma_h$
  end;
for $h := 1$ to s do
  if $\gamma_h = min$ then $\mathcal{T} := \mathcal{T} \cup \{h\}$;
$\mathcal{D}^*(app) := \mathcal{T}$.

The time requirements to obtain $\mathcal{D}^*(app)$ are negligible. The set that minimizes the light-traffic limit of $C(\mathcal{D})$ is not necessarily uniquely determined. More precisely, any non-empty subset of $\mathcal{T}$ yields the optimal value in (5.38). However, we propose to approximate $\mathcal{D}^*$ by $\mathcal{T}$, rather than by any proper non-empty subset of $\mathcal{T}$, because in the former case the proposed approximation is in agreement with the fact that in symmetrical models with exhaustive service at all queues, the mean amount of work in the system (i.e. $c_i = \rho_i/\rho$, $i = 1, \ldots, s$), is minimized only for the complete set $\mathcal{D}^* = \{1, \ldots, s\}$ (cf. [123]).

To check the quality of the heuristic approach presented above, we have computed $\mathcal{D}^*$ and $\mathcal{D}^*(app)$ for a fairly wide range of feasible model parameters. First, we consider a number of 3-queue models with the following set of parameters: $\boldsymbol{a} = (0.80, 0.40, 0.40)$; $\boldsymbol{\beta}^{(1)} = (0.50, 0.50, 1.00)$; $\boldsymbol{\sigma}^{(1)} = (2r, r, \frac{1}{2}r)$; the switch-over times are all Coxian distributed with squared coefficient of variation α; $\boldsymbol{c} = (5.00, 1.00, 1.00)$. Table 5.7 below shows the results for various values of α and r. The relative error, denoted by '$err\%$', is defined by

$$\frac{C(\mathcal{D}^*(app)) - C(\mathcal{D}^*)}{C(\mathcal{D}^*)} \times 100. \tag{5.45}$$

One may verify that we have $\mathcal{D}^*(app) = \{1\}$ in all considered cases.

To check the accuracy of the approximation for larger models, we also consider the following 6-queue model: $\boldsymbol{a} = (0.50, 0.10, 0.10, 0.10, 0.10)$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00, 1.00, 1.00, 1.00)$; $\boldsymbol{\sigma}^{(1)} = (0.50, 0.30, 0.50, 0.30, 0.50, 0.30)$. The switch-over times from Q_2 to Q_3, from Q_4 to Q_5 and from Q_6 to Q_1 are 2-phase Coxian distributed with squared coefficient of variation 10.00; all other switch-over times are exponentially distributed; $\boldsymbol{c} = (3/8, 1/8, 1/8, 1/8, 1/8, 1/8)$. Tables 5.8 and 5.9 show the results for various values of ρ, for $\boldsymbol{q} = (0.00, 0.00, 0.00)$ and $\boldsymbol{q} = (1.00, 1.00, 1.00)$, respectively. In this model we also have $\mathcal{D}^*(app) = \{1\}$.
The results in Tables 5.7 and 5.8 indicate that the proposed approximation yields fairly accurate results for a fairly broad range of models.

ρ	r	α	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	$C(\mathcal{D}^*(app))$	$err\%$
0.1	0.01	0.25	{1}	0.58	0.58	0.0
0.1	0.01	10.00	{1}	0.59	0.59	0.0
0.1	0.15	0.25	{1}	1.18	1.18	0.0
0.1	0.15	10.00	{1}	1.51	1.51	0.0
0.1	0.50	0.25	{1}	3.18	3.18	0.0
0.1	0.50	10.00	{1}	6.63	6.63	0.0
0.5	0.01	0.25	{1}	5.05	5.05	0.0
0.5	0.01	10.00	{1}	5.06	5.06	0.0
0.5	0.15	0.25	{1}	7.13	7.13	0.0
0.5	0.15	10.00	{1,2}	8.80	8.94	1.6
0.5	0.50	0.25	{1}	13.98	13.98	0.0
0.5	0.50	10.00	{1,2,3}	25.85	27.64	6.9
0.8	0.01	0.25	{1}	19.84	19.84	0.0
0.8	0.01	10.00	{1}	19.86	19.86	0.0
0.8	0.15	0.25	{1}	25.52	25.52	0.0
0.8	0.15	10.00	{1,2,3}	28.49	28.85	1.3
0.8	0.50	0.25	{1,2}	41.18	41.22	0.1
0.8	0.50	10.00	{1,2}	60.25	61.19	1.6

Table 5.7: Accuracy of the cost belonging to the approximated optima; $s = 3$.

Expression (5.38), and hence the approximated optimum $\mathcal{D}^*(app)$, only depends on the cost coefficients and on the mean switch-over times. So the approximated optimum does not depend on higher moments of the switch-over times, neither on the arrival rates, neither on the Bernoulli parameters. However, the optimal dormant set generally depends on higher moments of the variability of the switch-over times (cf. Table 5.7) and the arrival rates (cf. Table 5.6), and also on the Bernoulli parameters (cf. Tables 5.3 and 5.4). Hence, the accuracy of the approximation may become poor when the variability of the switch-over times becomes large (cf. Table 5.7) and when the arrival rates

	$\boldsymbol{q} = (0.00, 0.00, 0.00)$			
ρ	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	$C(\mathcal{D}^*(app))$	$err\%$
0.1	{ 1 }	1.48	1.48	0.0
0.3	{2,6}	5.99	5.99	0.1
0.5	-	∞	∞	-
0.7	-	∞	∞	-
0.8	-	∞	∞	-

Table 5.8: Exact and approximated optima; 1-limited service.

	$\boldsymbol{q} = (1.00, 1.00, 1.00)$			
ρ	$\mathcal{D}^*$	$C(\mathcal{D}^*)$	$C(\mathcal{D}^*(app))$	$err\%$
0.1	{1}	1.32	1.32	0.0
0.3	{1,3-6}	2.61	2.67	2.3
0.5	{1-6}	4.44	4.52	1.7
0.7	{1-6}	8.15	8.21	0.7
0.8	{1-6}	12.51	12.55	0.3

Table 5.9: Exact and approximated optima; exhaustive service.

are very asymmetrical.

The approximation (5.44) is based on the light-traffic limit of the mean waiting times at the queues (5.38), which in fact corresponds to the 0-th order term in the power-series expansions of the mean waiting times. Thus, the approximated optimum (5.44) can be considered as a *0-th order approximation* of the optimum $\mathcal{D}^*$. The accuracy of the approximated optimum may be considerably improved by taking higher-order approximations of $\mathcal{D}^*$. To this end, algebraic expressions for higher order terms of the power-series expansions of the mean waiting times should be obtained with the aid of the PSA (cf. (3.29)). Then, these expressions may be used to derive expressions for higher order approximations of the optimal dormant set.

Chapter 6

Polling systems with multiple servers

6.1 Introduction

The vast majority polling studies in the literature is devoted to *single*-server polling models. Polling models with more than one server have received very little attention, probably because of their mathematical complexity. So far, there are hardly any exact results known for these models, apart from some mean value results for global performance measures like cycle times and intervisit times. We analyze the performance of multiple-server polling models in which each of the servers visits the queues according to some fixed service order table. Such models appear to completely defy the derivation of any exact waiting-time results.
The aim of this chapter is to gain an insight into the performance of multiple-server polling systems with independent servers. The influence of the routing of the servers on the system performance is studied. We investigate the option of partitioning the system into a number of subsystems, each guided by particular servers. A comparison is made between the performance of multiple-server and single-server polling models with a comparable load.
Numerical experiments with

In the open literature a few papers have appeared about polling models with multiple servers. Borst [32] explores the class of models that allow an exact analysis in the case of *coupled* servers in which all servers visit the same queue at any time (while it is possible that some servers may idle at that queue when all customers present at that queue are being served by other servers). This class includes most single-queue models, two-queue two-server models with exhaustive service and exponential service times, as well as infinite server models with an arbitrary number of queues, exhaustive or gated service, and deterministic service times. For multiple-server polling models with coupled servers, Browne

and Weiss [53] obtain index rules for the minimization of the mean length of individual cycles for both the exhaustive and the gated service discipline. Browne et al. [51] derive the mean waiting time for a completely symmetric two-queue model with an infinite number of coupled servers and deterministic service times. Browne and Kella [52] obtain the busy-period distribution for a two-queue model with an infinite number of coupled servers, exhaustive service, and deterministic service times at one queue and general service times at the other queue.
In the case of *uncoupled* servers the class of models that allow an exact analysis is even smaller. Levy and Yechiali [124] and Kao and Narayanan [100] study the joint distribution of the queue lengths and the number of busy servers for a Markovian multiple-server queue, where the servers individually go on vacation when there are no waiting customers left. Mitrany and Avi-Itzhak [133] and Neuts and Lucantoni [137] analyze the joint distribution of the queue lengths and the number of busy servers for a Markovian multiple-server queue where servers break down at exponential intervals and then get repaired. For models with more than one queue with multiple uncoupled servers in which each server visits the queues according to some cyclic schedule, Morris and Wang [134] obtain the mean cycle time of each server and the mean intervisit time to a queue, and derive approximate expressions for the mean sojourn time for multiple-server variants of the gated and the limited service discipline. A very interesting phenomenon observed in [134] is the tendency of the servers to cluster if they follow identical routes, especially in heavy traffic. Numerical experiments indicate that the bunching of servers is likely to deteriorate the system performance. The bunching of servers is alleviated if they follow different routes. Therefore, Morris and Wang advocate the use of 'dispersive' schedules to improve the system performance. Motivated by these observations, Levy et al. [120] propose the so-called bang-bang policy. The main idea of this policy is that at each queue, the successively departing servers proceed in opposite directions. Numerical results show that the bang-bang policy performs significantly better than other policies over a wide range of parameters. However, for multiple-queue models with more than one independent server, there is not a single non-trivial model for which any exact results are known about more detailed performance measures such as the mean waiting times at the queues.

Because of the mathematical intractability of multiple-server polling models, a variety of mean waiting-time approximations has been developed. In references [14], [99], [101], [111], [139], [175] mean waiting-time approximations are developed to analyze the performance of Local Area Networks (LANs) with multiple token rings. A main disadvantage of these mean waiting-time approximations is that their accuracy degrades considerably when the clustering of the servers, as observed in [134], is significant. Mean waiting-time approximations oriented to LANs with a multiple-slotted ring are contained in references [14], [127], [165], [175], [177]. Ajmone Marsan et al. [1], [2], [4] derive the mean cycle time and bounds for the mean waiting times in symmetric models for the

exhaustive, gated and 1-limited service discipline. In [3] they illustrate how Petri-net techniques may be used to study Markovian multiple-server polling models. Gamse and Newell [90] obtain approximate expressions for the mean round-trip time for a multiple-elevator facility. They compare some control options of multiple parallel elevators and, although there are some distinguishing features in the model description, find a similar tendency to form bunches as observed by Morris and Wang [134].
The above-mentioned studies unanimously point out that multiple-server polling models, combining the complexity of single-server polling models and multiple-server models, are extraordinarily hard to analyze.

In this chapter we investigate the performance of polling models with multiple uncoupled servers, who visit the queues in some static order. To this end, we show how a broad class of multiple-queue polling models may be analyzed by means of the PSA. We investigate the tendency for the servers to bunch together. Numerical experiments indicate that this bunching generally deteriorates the system performance. We address the option of partitioning the system into a number of subsystems, each of which is attended by particular servers. It is shown that this option may become beneficial when the switch-over times are significant. In addition, we show that multiple-server models generally outperform single-server models which carry a comparable load. Finally, we propose a new mean waiting-time approximation, in which the clustering effect [134] is explicitly taken into account. Numerical experiments indicate that, as opposed to the existing mean waiting-time approximations in [14], [99], [101], [111], [139], [175], the proposed approximation yields accurate results for a wide variety of model parameters, also when the clustering is significant.

The remainder of this chapter is organized as follows. In section 6.2 we present a detailed model description. In section 6.3 we show how the present model can be described as a continuous-time Markov process and how this process can be analyzed by means of the PSA. The complexity of the PSA for the present model is discussed in section 6.4. In section 6.5 we give an overview of the numerical results. In section 6.6 we present a new mean waiting-time approximation. Section 6.7 contains some concluding remarks.

6.2 Model description

Consider the basic polling model (as discussed in section 1.3) consisting of s queues, $Q_1, \ldots, Q_s$, each of infinite capacity. Customers arrive at Q_i according to a Poisson process with rate $\lambda_i = a_i\rho$ and are referred to as type-i customers, $i = 1, \ldots, s$. The service times of type-i customers are exponentially distributed with mean $\beta_i^{(1)} = 1/\mu_i^1$, $i = 1, \ldots, s$. The customers are served by m *independent* servers. Server j visits the queues periodically according to a fixed service order table $\pi_j = (\pi_j(1), \ldots, \pi_j(L_j))$, where L_j is the (finite) length of the service order table for server j. That is, the l-th queue visited

by server j is $\pi_j((l-1) \bmod L_j)+1)$, $l = 1,2,\ldots,$ $j = 1,\ldots,m$. Define $\Pi_j := \{\pi_j(1),\ldots,\pi_j(L_j)\}$ to be the index set of the queues visited by server j, $j = 1,\ldots,m$. Note that the queues are not necessarily visited by each of the servers. The switch-over times needed by the servers to move from Q_i to Q_k are exponentially distributed with mean $\sigma_{i,k}^{(1)} = 1/\mu_{i,k}^0$, $i,k = 1,\ldots,s$. Denote by $\sigma_{i,j}^{(1)} = 1/\mu_{\pi_j(i-1),\pi_j(i)}^0$ the mean switch-over time needed by server j to move from $Q_{\pi_j(i-1)}$ to $Q_{\pi_j(i)}$, $i = 1,\ldots,L_j$, $j = 1,\ldots,m$. Denote by $\sigma_{1,j} := \Sigma_{i=1}^{L_j}\sigma_{i,j}^{(1)}$ the mean total switch-over time per 'cycle' of server j along the queues according to polling table $\boldsymbol{\pi}_j$, $j = 1,\ldots,m$.

The servers are assumed to visit the queues independently of each other, under the restriction that at most m_i servers may visit Q_i simultaneously. An arrival of server j ($j = 1,\ldots,m$) at Q_i will be called *effective* if server j finds less than m_i other servers working at Q_i and there are customers waiting at Q_i (so that server j may start serving at Q_i). If a server arrival is not effective, then the server immediately proceeds to the next queue according to its polling table. The service of a customer can not be interrupted.

The number of customers that is served during one visit of a server to a queue is determined a by Bernoulli schedule $\boldsymbol{q} = (q_1,\ldots,q_s)$ (cf. section 1.3). At each queue the queueing disciplines may be general, but may not depend on the actual service times. All service times, switch-over times, and interarrival times are assumed to be mutually independent and independent of the state of the system.

Finally some words on the stability conditions. Let $k_{i,j}$ be the number of times Q_i occurs in polling table $\boldsymbol{\pi}_j$, $j = 1,\ldots,m$, $i = 1,\ldots,s$. For $m = 1$, necessary and sufficient conditions are (cf. [85]): $\rho\left[1 + a_i\sigma_{1,1}(1-q_i)\right]/k_{i,1} < 1$. However, for the multiple-server case ($m > 1$), the stability conditions are generally not known. Evidently, necessary conditions are that $\rho_i < m_i$, $i = 1,\ldots,s$, and that for each set $I \subseteq \{1,\ldots,s\}$ the indices $i \in I$ occur in the polling table of at least $\Sigma_{i\in I}\rho_i$ servers (in particular for $I = \{1,\ldots,s\}$ implying $\rho < m$). We conjecture that these conditions are in fact also sufficient for $q_i = 1$, $i = 1,\ldots,s$, i.e. for the exhaustive service discipline. For $q_i < 1$, when the mean maximum number of customers served during a visit to Q_i is $1/(1-q_i)$, it is considerably harder to find the stability conditions. When $m_i = m$, $\sigma_{1,j} = \sigma_{1,1}$, $j = 1,\ldots,m$, $k_{i,j} = 1$, $i = 1,\ldots,s$, $j = 1,\ldots,m$, simple balancing arguments suggest that necessary and sufficient conditions are $\rho\left[1 + a_i\sigma_{1,1}(1-q_i)\right] < m$, $i = 1,\ldots,s$. Yet, in other cases with $q_i < 1$ in which these assumptions are not satisfied the problem of establishing the stability conditions appears to be completely open. Although they are not generally known, throughout this chapter the stability conditions are simply assumed to hold.

6.3 The power-series algorithm

In this section we will show how the present model can be analyzed by means of the PSA. In 6.3.1 the state probabilities are defined and the global balance

equations are formulated. In 6.3.2 the state probabilities are expressed as power series in the offered load to the system and we derive a complete computational scheme to calculate the coefficients of these power series.

6.3.1 Balance equations

Let $N_i(t)$ be the number of customers at Q_i (including customers in service) at time t, $i = 1, \ldots, s$, $t \geq 0$. Denote $\boldsymbol{N}(t) = (N_1(t), \ldots, N_s(t))$. Evidently, the joint queue-length process $\{\boldsymbol{N}(t), t \geq 0\}$ itself is not a Markov process, as the transitions also depend on the status of the servers. To extend the joint queue-length process to a Markov process, we introduce some supplementary variables describing the status of the servers. Let $H_j(t)$ be the actual entry in the polling table of server j at time t, $j = 1, \ldots, m$; let $Z_j(t)$ indicate whether server j is switching ($Z_j(t) = 0$) or serving ($Z_j(t) = 1$) at time t, $j = 1, \ldots, m$. So, if $(H_j(t), Z_j(t)) = (l, 0)$ then server j is switching to $Q_{\pi_j(l)}$ at time t; if $(H_j(t), Z_j(t)) = (l, 1)$ then server j is serving at queue $Q_{\pi_j(l)}$ at time t, $t \geq 0$. Denote $\boldsymbol{H}(t) = (H_1(t), \ldots, H_m(t))$, $\boldsymbol{Z}(t) = (Z_1(t), \ldots, Z_m(t))$. Define the supplementary space by $\mathcal{S} := \mathcal{S}_1 \times \mathcal{S}_2$, where

$$\mathcal{S}_1 := \{\boldsymbol{h} = (h_1, \ldots, h_m) \mid \; h_j \in \{1, \ldots, L_j\}, \; j = 1, \ldots, m\}, \tag{6.1}$$

$$\mathcal{S}_2 := \{\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_m) \mid \; \zeta_j \in \{0, 1\}, \; j = 1, \ldots, m\}, \tag{6.2}$$

with L_j the length of the polling table of server j, $j = 1, \ldots, m$. Then it is easily verified that the joint process $\{(\boldsymbol{N}(t), \boldsymbol{H}(t), \boldsymbol{Z}(t)), t \geq 0\}$ is a continuous-time Markov process with state space $\mathbb{N}^s \times \mathcal{S}$. We now derive the balance equations for the process $\{(\boldsymbol{N}(t), \boldsymbol{H}(t), \boldsymbol{Z}(t)), t \geq 0\}$. Denote by $(\boldsymbol{N}, \boldsymbol{H}, \boldsymbol{Z})$ stochastic variables with as joint distribution the joint stationary distribution of $(\boldsymbol{N}(t), \boldsymbol{H}(t), \boldsymbol{Z}(t))$.
For each state $(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$ denote the number of servers working at Q_i by: for $i = 1, \ldots, s$,

$$x_i(\boldsymbol{h}, \boldsymbol{\zeta}) := |\{j \mid \; (\pi_j(h_j), \zeta_j) = (i, 1)\}|. \tag{6.3}$$

The state probabilities are defined as follows: for $(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$,

$$p(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) := \Pr\{(\boldsymbol{N}, \boldsymbol{H}, \boldsymbol{Z}) = (\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta})\}. \tag{6.4}$$

The global balance equations for the present model read as follows: for $(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$, with $x_i(\boldsymbol{h}, \boldsymbol{\zeta}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, s$,

$$
\begin{aligned}
&\Bigg[\rho \sum_{i=1}^{s} a_i + \sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} I\{\zeta_j = 0\} \\
&\qquad\qquad + \sum_{j=1}^{m} \mu^1_{\pi_j(h_j)} I\{\zeta_j = 1\}\Bigg] p(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \\
&\rho \sum_{i=1}^{s} a_i p(\boldsymbol{n}-\boldsymbol{e}_i,\boldsymbol{h},\boldsymbol{\zeta}) I\{n_i > 0\} \\
&+ \sum_{j=1}^{m} \mu^1_{\pi_j(h_j)} p(\boldsymbol{n}+\boldsymbol{e}_{\pi_j(h_j)},\boldsymbol{h},\boldsymbol{\zeta}) q_{\pi_j(h_j)} I\{\zeta_j = 1\} \\
&+ \sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} p(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}-\boldsymbol{e}_j) I\{\zeta_j = 1\} \\
&+ \sum_{j=1}^{m} \mu^1_{\pi_j(h_j-1)} p(\boldsymbol{n}+\boldsymbol{e}_{\pi_j(h_j-1)},\boldsymbol{h}-\boldsymbol{e}_j,\boldsymbol{\zeta}+\boldsymbol{e}_j) \\
&\quad \times \left[1 - q_{\pi_j(h_j-1)} I\left\{x_{\pi_j(h_j-1)}(\boldsymbol{h},\boldsymbol{\zeta}) < n_{\pi_j(h_j-1)}\right\}\right] I\{\zeta_j = 0\} \\
&+ \sum_{j=1}^{m} \mu^0_{\pi_j(h_j-2),\pi_j(h_j-1)} p(\boldsymbol{n},\boldsymbol{h}-\boldsymbol{e}_j,\boldsymbol{\zeta}) \\
&\quad \times I\left\{x_{\pi_j(h_j-1)}(\boldsymbol{h},\boldsymbol{\zeta}) = \min\{n_{\pi_j(h_j-1)}, m_{\pi_j(h_j-1)}\}\right\} I\{\zeta_j = 0\}.
\end{aligned}
\tag{6.5}
$$

The first term at the right-hand side of (6.5) indicates an arrival at Q_i. The second term indicates that after a service completion of server j at $Q_{\pi_j(h_j)}$, server j starts to serve another customer at that queue. The third term corresponds to an effective arrival of server j at $Q_{\pi_j(h_j)}$ and a subsequent service initiation at that queue. The fourth term indicates that server j proceeds to the next queue after having completed a service at $Q_{\pi_j(h_j-1)}$. The fifth term indicates an arrival of server j at $Q_{\pi_j(h_j-1)}$ which is not effective, either because there are no customers waiting at that queue or because the maximal allowable number of servers is already working at that queue.

Because the number of servers that may be working at Q_i simultaneously is bounded by m_i and by n_i, we have: for $(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$,

$$
p(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = 0 \text{ if } \exists\, i \in \{1,\ldots,s\} \mid \; x_i(\boldsymbol{h},\boldsymbol{\zeta}) > \min\{n_i, m_i\}. \tag{6.6}
$$

In addition, the law of total probability implies

$$
\sum_{(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}} p(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = 1. \tag{6.7}
$$

6.3.2 Computational scheme

In this section we show how for any number of queues and any number of servers the PSA may be used to solve the set of global balance equations (6.5), (6.7). For the single-server case ($m = 1$) a complete computational scheme to calculate the coefficients is derived in [22]. In this section, we extend this computational scheme to the multiple-server case.

In section 2.3.2 conditions for the applicability of the PSA were given. The

conditions are satisfied if for each reachable state $(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$, $\boldsymbol{n} \neq \boldsymbol{0}$, the probability that a departure occurs before any arrival takes place is positive. It is readily verified that these conditions are satisfied in the present model. Based on the validity of these conditions for the present model, we have: for $(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$, $p(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) = O(\rho^{|\boldsymbol{n}|})$, $\rho \downarrow 0$ (cf. (2.5)). Based on this property, we introduce the following power-series expansions for the state probabilities: for $(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^s \times \mathcal{S}$, $x_i(\boldsymbol{h}, \boldsymbol{\zeta}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, s$,

$$p(\boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) = \rho^{|\boldsymbol{n}|} \sum_{k=0}^{\infty} \rho^k b_0(k; \boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}). \tag{6.8}$$

We now show how a computational scheme may be derived to calculate the coefficients of the power series. Substituting the power-series expansions (6.8) into the balance equations (6.5) and equating corresponding powers of ρ yields the following linear relations between the coefficients of the power series: for $(k; \boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ with $x_i(\boldsymbol{h}, \boldsymbol{\zeta}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, s$,

$$\begin{aligned}
&\Bigg[\sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} I\{\zeta_j = 0\} \\
&\qquad + \sum_{j=1}^{m} \mu^1_{\pi_j(h_j)} I\{\zeta_j = 1\}\Bigg] b_0(k; \boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) = \\
&\sum_{i=1}^{s} a_i \left[b_0(k; \boldsymbol{n} - \boldsymbol{e}_i, \boldsymbol{h}, \boldsymbol{\zeta}) I\{n_i > 0\} - b_0(k-1; \boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) I\{k > 0\}\right] \\
&+ \sum_{j=1}^{m} \mu^1_{\pi_j(h_j)} b_0(k-1; \boldsymbol{n} + \boldsymbol{e}_{\pi_j(h_j)}, \boldsymbol{h}, \boldsymbol{\zeta}) q_{\pi_j(h_j)} I\{\zeta_j = 1\} I\{k > 0\} \\
&+ \sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} b_0(k; \boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta} - \boldsymbol{e}_j) I\{\zeta_j = 1\} \\
&+ \sum_{j=1}^{m} \mu^1_{\pi_j(h_j-1)} b_0(k-1; \boldsymbol{n} + \boldsymbol{e}_{\pi_j(h_j-1)}, \boldsymbol{h} - \boldsymbol{e}_j, \boldsymbol{\zeta} + \boldsymbol{e}_j) \\
&\quad \times \left[1 - q_{\pi_j(h_j-1)} I\{x_{\pi_j(h_j-1)}(\boldsymbol{h}, \boldsymbol{\zeta}) < n_{\pi_j(h_j-1)}\}\right] \\
&\quad \times I\{\zeta_j = 0\} I\{k > 0\} \\
&+ \sum_{j=1}^{m} \mu^0_{\pi_j(h_j-2),\pi_j(h_j-1)} b_0(k; \boldsymbol{n}, \boldsymbol{h} - \boldsymbol{e}_j, \boldsymbol{\zeta}) \\
&\quad \times I\left\{x_{\pi_j(h_j-1)}(\boldsymbol{h}, \boldsymbol{\zeta}) = \min\{n_{\pi_j(h_j-1)}, m_{\pi_j(h_j-1)}\}\right\} I\{\zeta_j = 0\}.
\end{aligned} \tag{6.9}$$

By rearranging the terms at the right-hand side, the set of equations (6.9) can be rewritten as follows: for $(k; \boldsymbol{n}, \boldsymbol{h}, \boldsymbol{\zeta}) \in \mathbb{N}^{1+s} \times \mathcal{S}$, with $x_i(\boldsymbol{h}, \boldsymbol{\zeta}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, s$,

$$\begin{aligned}
&\Bigg[\sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} I\{\zeta_j = 0\} \\
&\qquad\qquad + \sum_{j=1}^{m} \mu^1_{\pi_j(h_j)} I\{\zeta_j = 1\}\Bigg] b_0(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \\
&\sum_{j=1}^{m} \mu^0_{\pi_j(h_j-2),\pi_j(h_j-1)} b_0(k;\boldsymbol{n},\boldsymbol{h}-\boldsymbol{e}_j,\boldsymbol{\zeta}) \\
&\quad \times I\left\{x_{\pi_j(h_j-1)}(\boldsymbol{h},\boldsymbol{\zeta}) = \min\{n_{\pi_j(h_j-1)}, m_{\pi_j(h_j-1)}\}\right\} I\{\zeta_j = 0\} \\
&+ y_0(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}),
\end{aligned} \tag{6.10}$$

where

$$\begin{aligned}
&y_0(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) := \\
&\sum_{i=1}^{s} a_i \left[b_0(k;\boldsymbol{n}-\boldsymbol{e}_i,\boldsymbol{h},\boldsymbol{\zeta}) I\{n_i > 0\} - b_0(k-1;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) I\{k > 0\}\right] \\
&+ \sum_{j=1}^{m} \mu^1_{\pi_j(h_j)} b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_{\pi_j(h_j)},\boldsymbol{h},\boldsymbol{\zeta}) q_{\pi_j(h_j)} I\{\zeta_j = 1\} I\{k > 0\} \\
&+ \sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} b_0(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}-\boldsymbol{e}_j) I\{\zeta_j = 1\} \\
&+ \sum_{j=1}^{m} \mu^1_{\pi_j(h_j-1)} b_0(k-1;\boldsymbol{n}+\boldsymbol{e}_{\pi_j(h_j-1)},\boldsymbol{h}-\boldsymbol{e}_j,\boldsymbol{\zeta}+\boldsymbol{e}_j) \\
&\quad \times \left[1 - q_{\pi_j(h_j-1)} I\{x_{\pi_j(h_j-1)}(\boldsymbol{h},\boldsymbol{\zeta}) < n_{\pi_j(h_j-1)}\}\right] \\
&\quad \times I\{\zeta_j = 0\} I\{k > 0\}.
\end{aligned} \tag{6.11}$$

We will now show how the relations (6.10), (6.11) can be used to compute the coefficients $b_0(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$ mainly recursively. To this end, we first define the following partial ordering of the vectors $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$ (cf. also (2.12)): for $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}), (\hat{k};\hat{\boldsymbol{n}},\hat{\boldsymbol{h}},\hat{\boldsymbol{\zeta}}) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$\begin{aligned}
&(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \prec (\hat{k};\hat{\boldsymbol{n}},\hat{\boldsymbol{h}},\hat{\boldsymbol{\zeta}}) \\
&\text{if } \left[k+|\boldsymbol{n}| < \hat{k}+|\hat{\boldsymbol{n}}|\right] \vee \left[k+|\boldsymbol{n}| = \hat{k}+|\hat{\boldsymbol{n}}| \wedge k < \hat{k}\right].
\end{aligned} \tag{6.12}$$

Next, we extend the partial ordering $\prec$ to the vectors of supplementary values $(\boldsymbol{h},\boldsymbol{\zeta})$: for $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$, $(k;\boldsymbol{n},\hat{\boldsymbol{h}},\hat{\boldsymbol{\zeta}}) \in \mathbb{N}^{1+s} \times \mathcal{S}$,

$$\begin{aligned}
&(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \prec (k;\boldsymbol{n},\hat{\boldsymbol{h}},\hat{\boldsymbol{\zeta}}) \\
&\text{if } \left[\forall j = 1,\ldots,m : \zeta_j \le \hat{\zeta}_j\right] \wedge \left[\exists j^* : \zeta_{j^*} < \hat{\zeta}_{j^*}\right].
\end{aligned} \tag{6.13}$$

It is readily verified that all coefficients in the right-hand side of (6.11) are of lower order than $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$ with respect to $\prec$ and hence, may be considered to be known in (6.10). So, for *given* k, $\boldsymbol{n}$ and $\boldsymbol{\zeta}$, it remains to define an ordering of the vectors $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$, $\boldsymbol{h} \in \mathcal{S}_1$. For given $\boldsymbol{\zeta} \in \mathcal{S}_2$, we partition the index set $\{1,\ldots,m\}$ into the following two subsets (corresponding to the collections of switching and serving servers, respectively):

$$\mathcal{C}^{(0)}(\boldsymbol{\zeta}) := \{j \mid \zeta_j = 0\}, \qquad \mathcal{C}^{(1)}(\boldsymbol{\zeta}) := \{j \mid \zeta_j = 1\}. \tag{6.14}$$

To derive an ordering for the vectors $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$, $\boldsymbol{h} \in \mathcal{S}_1$, for given values of k, $\boldsymbol{n}$ and $\boldsymbol{\zeta}$, it should be noted that the right-hand side of (6.10) motivates to partition the index set $\mathcal{C}^{(0)}(\boldsymbol{\zeta})$ (for fixed $\boldsymbol{n}$, $\boldsymbol{h}$ and $\boldsymbol{\zeta}$) into the following two subsets:

$$\begin{aligned} &\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) := \\ &\left\{ j \in \mathcal{C}^{(0)}(\boldsymbol{\zeta}) \mid \ x_i(\boldsymbol{h},\boldsymbol{\zeta}) = \min\{n_i, m_i\} \text{ for all } i \in \Pi_j \right\}, \end{aligned} \tag{6.15}$$

i.e. the set of switching servers that can not start serving at a queue as long as neither arrivals nor service completions occur; and

$$\begin{aligned} &\mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) := \mathcal{C}^{(0)}(\boldsymbol{\zeta}) \setminus \mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \\ &= \left\{ j \in \mathcal{C}^{(0)}(\boldsymbol{\zeta}) \mid \ \exists i^* \in \Pi_j : \ x_{i^*}(\boldsymbol{h},\boldsymbol{\zeta}) < \min\{n_{i^*}, m_{i^*}\} \right\}, \end{aligned} \tag{6.16}$$

i.e. the set of switching servers that *can* start serving at a queue (e.g. Q_{i^*}), even before either an arrival, or a service completion, or a switch-over completion of another server occurs.

We now distinguish, for *given* k, $\boldsymbol{n}$, $\boldsymbol{\zeta}$ *and* h_j $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, between two cases: (1) $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$, and (2) $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \neq \emptyset$.

Case 1: $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$.
We show how a recursive computational scheme for the coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, can be accomplished in this case. From $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$ it follows that there exists an $\boldsymbol{h}^* \in \mathcal{S}_1$, with $h_j^* = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, such that for any $j \in \mathcal{C}^{(0)}(\boldsymbol{\zeta})$ there exists an $i^* \in \Pi_j$ (namely, $i^* = \pi_j(h_j^* - 1)$) for which $x_{i^*}(\boldsymbol{h}^*,\boldsymbol{\zeta}) = x_{i^*}(\boldsymbol{h},\boldsymbol{\zeta}) < \min\{n_{i^*}, m_{i^*}\}$. Then the first term at the right-hand side of (6.10) vanishes, so that $b_0(k;\boldsymbol{n},\boldsymbol{h}^*,\boldsymbol{\zeta})$ is expressed only in terms of lower order w.r.t. $\prec$, cf. (6.12)-(6.13). The coefficient $b_0(k;\boldsymbol{n},\boldsymbol{h}^*,\boldsymbol{\zeta})$ will be used as starting point for a recursive computation of the coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j = h_j^*$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$. To this end, we define the following ordering of the vectors $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$: for $(k;\boldsymbol{n},\boldsymbol{h}',\boldsymbol{\zeta}),(k;\boldsymbol{n},\boldsymbol{h}'',\boldsymbol{\zeta}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ (with $h_j' = h_j'' = h_j^* = h_j$, $j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta})$),

$$\begin{aligned} &(k;\boldsymbol{n},\boldsymbol{h}',\boldsymbol{\zeta}) \prec (k;\boldsymbol{n},\boldsymbol{h}'',\boldsymbol{\zeta}) \text{ if} \\ &\left[\forall j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) : (h_j' - h_j^*) \bmod L_j \leq (h_j'' - h_j^*) \bmod L_j\right] \wedge \\ &\left[\exists j^* \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) : (h_{j^*}' - h_{j^*}^*) \bmod L_{j^*} < (h_{j^*}'' - h_{j^*}^*) \bmod L_{j^*}\right]. \end{aligned} \tag{6.17}$$

As an illustration of the ordering defined in (6.17), consider the following parameters: $m = 4$, $L_1 = 3$, $L_2 = 2$, $L_3 = 3$, $L_4 = 2$ and $\boldsymbol{\zeta} = (1,0,0,1)$. Then we have $\mathcal{C}^{(0)}(\boldsymbol{\zeta}) = \{2,3\}$, $\mathcal{C}^{(1)}(\boldsymbol{\zeta}) = \{1,4\}$. If $h_1 = 2$, $h_4 = 1$, and $\boldsymbol{h}^* = (2,2,2,1)$, then the vectors $\boldsymbol{h} \in \mathcal{S}_1$, with given $h_1 = h_1^*$, $h_4 = h_4^*$, are ranked in increasing order as first $(2,2,2,1)$, then $(2,2,3,1)$, $(2,1,2,1)$, then $(2,2,1,1)$, $(2,1,3,1)$, and finally $(2,1,1,1)$.

Case 2: $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \neq \emptyset$.
In this case the first term at the right-hand side of (6.10) does not vanish, so that the coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, can not be calculated recursively from (6.10) and (6.11). By definition, for each $j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$ there exist an $i^* \in \Pi_j$, and an $h_j^* \in \{1,\ldots,L_j\}$ with $i^* = \pi_j(h_j^* - 1)$, for which $x_{i^*}(\boldsymbol{h},\boldsymbol{\zeta}) < \min\{n_{i^*}, m_{i^*}\}$. Hence, the coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$ *and* $\hat{h}_j = h_j^*$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$, can be computed by solving the set of $\prod_{j\in\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})} L_j$ linear equations induced by (6.10). Then, the $\prod_{j\in\mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})} L_j$ coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, can be computed by solving the set of equations (6.10) for the coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, $\hat{h}_j = \tilde{h}_j$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$ in increasing order of the values of the combinations $\tilde{h}_j$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$ w.r.t. the partial ordering defined in (6.17), starting with $\tilde{h}_j = h_j^*$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$. It should be noted that the determinant of the set of equations for the coefficients $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$ and $\hat{h}_j = h_j^*$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$, is the same for each given $k,\boldsymbol{n},h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$ and $\hat{h}_j = h_j^*(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$, so that the number of matrix inversions can be reduced significantly. We refer to the end of this section for a more intuitive characterization of whether or not (6.10) is recursively solvable.

The same conditions which guarantee that (6.8) holds also guarantee that these sets possess a unique solution (cf. [22]), except for the case $\boldsymbol{n} = \boldsymbol{0}$. So the only states that need further attention are the states with $\boldsymbol{n} = \boldsymbol{0}$, and hence with $\boldsymbol{\zeta} = \boldsymbol{0}$ (because $p(\boldsymbol{0},\boldsymbol{h},\boldsymbol{\zeta}) = 0$ if $\boldsymbol{\zeta} \neq \boldsymbol{0}$, cf. (6.6)). In this case the set of equations (6.10) reads: for $(k;\boldsymbol{h}) \in \mathbb{N} \times \mathcal{S}_1$,

$$\begin{aligned}&\sum_{j=1}^{m} \mu^0_{\pi_j(h_j-1),\pi_j(h_j)} b_0(k;\boldsymbol{0},\boldsymbol{h},\boldsymbol{0}) = \\ &\sum_{j=1}^{m} \mu^0_{\pi_j(h_j-2),\pi_j(h_j-1)} b_0(k;\boldsymbol{0},\boldsymbol{h}-\boldsymbol{e}_j,\boldsymbol{0}).\end{aligned} \tag{6.18}$$

One may verify by summing the equations (6.18) over $\boldsymbol{h}$, $\boldsymbol{h} \in \mathcal{S}_1$, that this set of equations is dependent. However, the law of total probability (6.7) yields (together with (6.8)) the following additional equation: for $k = 0,1,\ldots,$

$$\sum_{(\boldsymbol{h},\boldsymbol{\zeta})\in\mathcal{S}} b_0(k;\boldsymbol{0},\boldsymbol{h},\boldsymbol{\zeta}) = Y_0(k), \tag{6.19}$$

where $Y_0(0) := 1$ and for $k = 1,2,\ldots,$

$$Y_0(k) := -\sum_{0<|\boldsymbol{n}|\leq k} \sum_{(\boldsymbol{h},\boldsymbol{\zeta})\in\mathcal{S}} b_0(k-|\boldsymbol{n}|;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}). \tag{6.20}$$

Note that the right-hand side of (6.20) contains only coefficients of lower order w.r.t. $\prec$ than $(k;\mathbf{0},\boldsymbol{h},\boldsymbol{\zeta})$, cf. (6.12). Now, all but one of the equations in (6.18) together with either (6.19) or (6.20) uniquely determine the coefficients $b_0(k;\mathbf{0},\boldsymbol{h},\mathbf{0})$, $\boldsymbol{h} \in \mathcal{S}_1$, for $k = 0,1,\ldots$, provided the Markov process $\{(\boldsymbol{N}(t),\boldsymbol{H}(t),\boldsymbol{\zeta}(t)), t \geq 0\}$, conditioned on $\boldsymbol{N}(t) = 0$ and $\boldsymbol{\zeta}(t) = 0$, is irreducible. This condition is satisfied if in the case $\rho = 0$ each state $(\mathbf{0},\boldsymbol{h},\mathbf{0})$, $\boldsymbol{h} \in \mathcal{S}_1$, can be reached from any other state with $(\mathbf{0},\hat{\boldsymbol{h}},\mathbf{0})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, i.e. if the $\mathbf{0}$-process (cf. section 2.3.3) is irreducible. For the present model this condition is satisfied, because in the $\mathbf{0}$-process the servers keep on switching along the queues according to their respective service order tables in a periodic way, independently of each other.

We will now present a computational scheme to calculate all of the coefficients $b_0(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$, with $(k;\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \in \mathbb{N}^{1+s} \times \mathcal{S}$ (with $x_i(\boldsymbol{h},\boldsymbol{\zeta}) \leq \min\{n_i, m_i\}$, $i = 1,\ldots,s$) up to the M-th power of ρ.

step 1 : $m := 0$;

step 2 : for all $(k;\boldsymbol{n}) \in \mathbb{N}^{1+s}$ with $\boldsymbol{n} \neq \mathbf{0}$ and with $k + |\boldsymbol{n}| = m$ do
for all $\boldsymbol{\zeta} \in \mathcal{S}_2$ (in increasing order of $\boldsymbol{\zeta}$ w.r.t. $\prec$ (cf. (6.12), (6.13))) do
for all possible combinations of h_j $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$ do
if $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$ then
determine $\boldsymbol{h}^*$ as follows:
for $j \in \mathcal{C}^{(0)}(\boldsymbol{\zeta})$, determine $i^* \in \Pi_j$ for which $x_{i^*}(\boldsymbol{h},\boldsymbol{\zeta}) < \min\{n_{i^*}, m_{i^*}\}$, and determine h_j^* such that $i^* = \pi_j(h_j^* - 1)$;
for $j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta})$, let $h_j^* = h_j$;
determine $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, *recursively* in increasing order of $\prec$ (cf. (6.17));
if $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) \neq \emptyset$ then
determine $\boldsymbol{h}^*$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$ as follows:
determine $i^* \in \Pi_j$ for which $x_{i^*}(\boldsymbol{h},\boldsymbol{\zeta}) < \min\{n_{i^*}, m_{i^*}\}$, and determine h_j^* such that $i^* = \pi_j(h_j^* - 1)$;
compute the $b_0(k;\boldsymbol{n},\hat{\boldsymbol{h}},\boldsymbol{\zeta})$, $\hat{\boldsymbol{h}} \in \mathcal{S}_1$, with $\hat{h}_j = h_j$ $(j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta}))$, by successively solving the set of linear equations (6.10) for $\hat{h}_j = \tilde{h}_j$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$ in increasing order of the combinations $\tilde{h}_j$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$ with respect to (6.17) (starting with $\tilde{h}_j = h_j^*$ $(j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}))$);

step 3 : determine $b_0(m;\mathbf{0},\boldsymbol{h},\mathbf{0})$, $\boldsymbol{h} \in \mathcal{S}_1$, by solving the set of equations (6.18) together with either (6.19) or (6.20);

step 4 : $m := m + 1$; if $m \leq M$ then return to *step 2*; otherwise STOP.

General performance measures of the form $\mathrm{E}\left\{g^{(l)}(\boldsymbol{N},\boldsymbol{H},\boldsymbol{Z})\right\}$, $l = 1,\ldots,L$, can be determined along the same lines as discussed in chapters 2 and 3.

The assumption of exponentially distributed service and switch-over times mainly serves the ease of the presentation. In fact, the approach presented in this section can be generalized in a straightforward manner to models with Coxian distributed service times and switch-over times along the same lines as in chapters 3, 4 and 5.

The approach presented in this section can also readily be extended to multiple-server models in which some of the switch-over times are negligible (i.e. $\mu^0_{i,k} = \infty$). When each server incurs at least one strictly positive switch-over time on a tour along the queues, only some slight modifications of the balance equations have to be made. On the other hand, when one or more servers do not incur any positive switch-over times, additional information has to be specified (for instance, as to which of the non-busy servers serves the first arriving customer, and at which entry of its polling table) to derive the global balance equations.

In chapter 2 we presented a general approach to compute derivatives with respect to a class of continuous system parameters. Clearly, the same approach can be applied for the present model, opening possibilities for performing sensitivity analysis and for studying optimization of the system with respect to the system parameters.

To characterize whether or not the set of equations (6.10) is recursively solvable, let us reconsider the set of equations (6.10) for *given* values of $\boldsymbol{k}$, $\boldsymbol{n}$, $\boldsymbol{\zeta}$ and h_j ($j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta})$). That is, we consider the following information on the current state of the system to be known: (i) the joint queue length, (ii) whether each of the servers is serving or switching, and (iii) the queues (entries) at which the serving servers are working. Then, given this configuration, $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$ is the set of indices corresponding to *those* switching servers j that have to skip *each* queue ($i = \pi_j(h_j - 1)$) that they visit, either because the maximal allowable number of servers is already working at that queue (i.e. $x_i(\boldsymbol{h},\boldsymbol{\zeta}) = m_i$) or because there are no waiting customers at that queue (i.e. $x_i(\boldsymbol{h},\boldsymbol{\zeta}) = n_i$). So, as long as neither arrivals nor service completions occur, the server j ($j \in \mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$) will keep on moving along the queues without serving any customer. Thus, for *given* $\boldsymbol{k}$, $\boldsymbol{n}$, $\boldsymbol{\zeta}$ and h_j ($j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta})$), the set of equations (6.10) is not recursively solvable if and only if one or more servers will keep on moving along the queues (according to their respective polling tables) as long as no arrivals nor service completions occur.
In the case that all polling tables are surjective mappings, i.e. each queue is visited by each of the servers, then (for given values of $\boldsymbol{k}$, $\boldsymbol{n}$, $\boldsymbol{\zeta}$ and h_j ($j \in \mathcal{C}^{(1)}(\boldsymbol{\zeta})$)) either $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$ (and $\mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \mathcal{C}^{(0)}(\boldsymbol{\zeta})$) or $\mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$ (and $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \mathcal{C}^{(0)}(\boldsymbol{\zeta})$). To see this, suppose $\mathcal{C}_A^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}) = \emptyset$, then there ex-

ists an $i \in \{1,\ldots,s\}$ for which $x_i(\boldsymbol{h},\boldsymbol{\zeta}) < \min\{n_i, m_i\}$. Because each server visits each of the queues, for each $j \in \mathcal{C}^{(0)}(\boldsymbol{\zeta})$ there exists an $\tilde{h}_j$ for which $\pi_j(\tilde{h}_j - 1) = i$, so that $j \in \mathcal{C}_B^{(0)}(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta})$. As a result, as long as neither arrivals nor service completions occur, either all switching servers or none of them will keep on moving along the queues without serving any customers.

The approach discussed here is also readily applicable to polling models in which the servers are coupled, i.e. all servers visit the same queue at any time. In that case, the set of feasible values for the supplementary variable $\boldsymbol{H}(t)$ is reduced to $\mathcal{S}_1 := \{1,\ldots,L_1\}$, where L_1 is the length of the (single) service order table of the coupled servers. It should be noted here that some of the servers will be idle in case the number of customers at a queue which is being served by the coupled servers is smaller than the number of servers.

6.4 Complexity

The time and memory requirements of the PSA increase exponentially with the number of queues, so that its use is restricted to models with a fairly small number of queues, cf. e.g. [22]. The total number of terms that has to be evaluated to compute the coefficients of the power series up to the M-th power of ρ is given by

$$\left(\begin{array}{c} M+s+1 \\ s+1 \end{array} \right) \times \prod_{j=1}^{m} L_j \times 2^m, \tag{6.21}$$

where the first factor indicates the number of vectors $(k;\boldsymbol{n})$ for which $k+|\boldsymbol{n}| \leq M$; the second and third factor together indicate the size of the supplementary space (cf. also section 2.3.5). Thus, the required amounts of computation time and storage capacity also increase *exponentially* in the number of servers.

Because in practice only a limited number of performance measures have to be evaluated (e.g. mean waiting times) rather than all individual state probabilities, the coefficients of the state probabilities can be removed as soon as they are not needed anymore in further computations. Then, the coefficients of the power-series expansions of the important performance measures can be aggregated during the execution of the PSA (cf. (2.18), (2.19)), and stored in (relatively small) arrays. This approach restricts the storage requirements to (cf. also section 2.3.5)

$$\left(\begin{array}{c} M+s \\ s \end{array} \right) \times \prod_{j=1}^{m} L_j \times 2^m. \tag{6.22}$$

As an illustration, consider a model in which all servers move along the queues in a strictly cyclic manner (so that $L_j = s$, $j = 1,\ldots,m$). Table 6.1 gives the maximal number of coefficients of the power series that can be computed

according to (6.22) for given amounts of storage capacity and for various values of the number of servers and the number of queues.

		10^6 coefficients				10^7 coefficients			
$s \rightarrow$		2	3	4	5	2	3	4	5
m	1	705	98	39	33	2234	213	71	38
$\downarrow$	2	352	53	22	13	1116	116	41	23
	3	175	28	12	7	557	63	23	13
	4	86	14	6	3	278	33	13	7
	5	42	7	2	1	138	17	6	3

Table 6.1: Maximal number of terms for different amounts of memory space.

Table 6.1 illustrates that the number of terms of the power series that can be computed for a given amount of storage capacity may decrease considerably when the numbers of servers and queues are increased. We refer to section 2.3.5 for some rough guidelines on the number of terms needed to achieve an acceptable degree of accuracy.

6.5 Numerical results

In this section we give an overview of the numerical results that we have gathered. We investigate the tendency of the servers to cluster and the influence of the visit orders on the system performance. We also consider the option of partitioning a multiple-server system into separate systems. Finally, we make some comparisons between multiple-server models and single-server models carrying a comparable load.
In the numerical examples the number of terms of the power series that has been computed depends on the offered load, and varies from $M = 15$ (for $\rho/m = 0.3$) to $M = 40$ (for $\rho/m = 0.95$). In all cases, the estimated accuracy (cf. section 2.3.4) is in the order of magnitude of 0.001.

Coalescing of the servers; influence of the visit orders
An interesting property of multiple-server polling models is the fact that the servers tend to coalesce, especially in heavily loaded systems in which the servers follow the same route. This phenomenon may be visualized as follows. A trailing server will tend to move fast, as it only encounters recently served queues, whereas a leading server will tend to be slowed down by queues that have not been served for a while, so that the servers tend to form bunches while constantly leapfrogging over another. This explains why the visit orders and the system load play a role in this coalescing effect.
To illustrate this, consider a model with the following parameters: $s = 4$; $m = 2$; $\boldsymbol{a} = (0.25, 0.25, 0.25, 0.25)$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00, 1.00)$; $m_i = 2$, $i = 1, \ldots, 4$; $q_i = 1.00$, $i = 1, \ldots, 4$ (i.e. exhaustive service); $\sigma_{i,k}^{(1)} = \alpha$, $i, k =$

$1,\ldots,4$, with α to be specified later on; $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = (1,2,3,4)$. We define the joint probability distribution of the server positions as follows:

$$P(k_1,\ldots,k_m) := \Pr\{\text{server } j \text{ is working at or switching to } Q_{k_j},\ j=1,\ldots,m\}, \tag{6.23}$$

for $k_j = 1,\ldots,s,\ j = 1,\ldots,m$. The joint server-position distribution can be expressed in terms of the state probabilities as follows: for $k_j = 1,\ldots,s,\ j = 1,\ldots,m$,

$$P(k_1,\ldots,k_m) = \sum_{\boldsymbol{n}\in\mathbb{N}^s} \sum_{\boldsymbol{h}\in\mathcal{S}_1:\pi_j(h_j)=k_j,\ j=1,\ldots,m} \sum_{\boldsymbol{\zeta}\in\mathcal{S}_2} p(\boldsymbol{n},\boldsymbol{h},\boldsymbol{\zeta}). \tag{6.24}$$

For $m = 2$, define $\mathbf{P} := (P(k_1,k_2))$ to be the matrix of joint server-position probabilities. Table 6.2 below shows the mean waiting times at the queues EW (which are equal for each queue) and the server-position distribution $P(k_1,k_2)$ for $\alpha = 0.05$ and for $\rho = 0.4$, 1.6 and 1.9.

$\rho = 0.4$		$\rho = 1.6$		$\rho = 1.9$	
EW	$\mathbf{P}$ (×100)	EW	$\mathbf{P}$ (×100)	EW	$\mathbf{P}$ (×100)
0.14	**6.3** 6.2 6.2 6.2	2.08	**7.8** 5.9 5.4 5.9	10.68	**14.6** 3.9 2.5 3.9
	6.2 **6.3** 6.2 6.2		5.9 **7.8** 5.9 5.4		3.9 **14.6** 3.9 2.5
	6.2 6.2 **6.3** 6.2		5.4 5.9 **7.8** 5.9		2.5 3.9 **14.6** 3.9
	6.2 6.2 6.2 **6.3**		5.9 5.4 5.9 **7.8**		3.9 2.4 3.9 **14.6**

Table 6.2: Coalescing effect.

Table 6.2 illustrates that the tendency of the servers to cluster increases as the traffic intensity grows. Indeed, if the system is lightly loaded, the switch-over times tend to predominate the lengths of the visit periods and will thus constantly disperse the servers over the system. On the other hand, in heavily-loaded systems the visit periods will dominate the switch-over times and drive the servers together.

To investigate the impact of the visit orders on the system performance, we have computed the mean waiting times at the various queues, EW_i (which are no longer equal for each queue when the visit orders of the servers are different), and EV, where V stands for the total amount of unfinished work in the system. Note that EV is related to the mean waiting times as $EV = \Sigma_{i=1}^{s}\left[\rho_i EW_i + \frac{1}{2}\lambda_i\beta_i^{(2)}\right]$ (cf. [55]). The model considered here has the same parameters as before, but has different visit orders. As before $\boldsymbol{\pi}_1 = (1,2,3,4)$, but $\boldsymbol{\pi}_2$ is varied over all possible permutations of the index set $\{1,2,3,4\}$. Note that because of the symmetry of the model there are only three non-equivalent cyclic service order combinations. Table 6.3 shows the mean waiting times $E\boldsymbol{W} = (EW_1, EW_2, EW_3)$ and the value of EV for $\alpha = 0.00$, 0.05 and 0.25, for

$\rho = 1.6$ and 1.8, respectively. Note that the case $\alpha = 0.00$ corresponds to a model with zero switch-over times, cf. the remarks at the end of section 6.3.2.

		$\rho = 1.6$		$\rho = 1.8$	
α	π_2	E$\boldsymbol{W}$	EV	E$\boldsymbol{W}$	EV
0.00	(1,2,3,4)	(1.78,1.78,1.78,1.78)	4.44	(4.26,4.26,4.26,4.26)	9.47
	(1,2,4,3)	(1.78,1.85,1.74,1.74)	4.44	(4.42,4.67,3.98,3.98)	9.47
	(1,4,3,2)	(1.78,1.78,1.78,1.78)	4.44	(4.26,4.26,4.26,4.26)	9.47
0.05	(1,2,3,4)	(2.08,2.08,2.08,2.08)	4.92	(4.87,4.87,4.87,4.87)	10.56
	(1,2,4,3)	(2.07,2.15,2.02,2.02)	4.91	(5.00,5.27,4.47,4.47)	10.45
	(1,4,3,2)	(2.06,2.06,2.06,2.06)	4.90	(4.79,4.79,4.79,4.79)	10.43
0.25	(1,2,3,4)	(3.29,3.29,3.29,3.29)	6.87	(7.36,7.36,7.36,7.36)	15.05
	(1,2,4,3)	(3.24,3.40,3.09,3.09)	6.73	(7.52,7.87,6.39,6.39)	14.48
	(1,4,3,2)	(3.16,3.16,3.16,3.16)	6.67	(6.86,6.86,6.86,6.86)	14.15

Table 6.3: Influence of the visit order.

Table 6.3 shows that the service orders may have a considerable impact on the individual mean waiting times. For single-server models similar observations have been made by Blanc [19]. However, for single-server models with exhaustive service it is well-known that EV is completely insensitive to the service order (as long as it is strictly cyclic), cf. Boxma and Groenendijk [40]. Table 6.3 suggests that in multiple-server models EV is perhaps not extremely sensitive to the service orders but definitely not completely insensitive. In fact, also the individual mean waiting times appear to be more sensitive to the service orders in multiple-server models. Tables 6.3 shows e.g. that in a multiple-server model even in case of a completely symmetric configuration the individual mean waiting times depend on the service order, unlike in a single-server model. The difference in sensitivity may be intuitively explained as follows. In case of exhaustive service the individual mean waiting times strongly depend on the mean residual intervisit time. In a single-server model the intervisit time is the time needed for the server *itself* to reach the queue again, i.e. the time involved in passing through the complete system once, which usually only marginally depends on the service order. In a multiple-server model the intervisit time is the time for *any* of the servers to reach the queue again, which strongly varies with the degree of clustering as implied by the service order. Table 6.3 points out e.g. that $\pi_2 = (1, 4, 3, 2)$ yields the best global performance (i.e. minimal EV) in all considered cases. Indeed, $\pi_2 = (1, 4, 3, 2)$ is likely to minimize the degree of clustering. The latter observation is in line with the observation of Morris and Wang [134] that the system performance can be improved when the coalescing effect is alleviated by using dispersive schedules. Note that in the case $\alpha = 0$, because of the symmetry of the model, EV does not depend on the service orders and has the same distribution as in a classical $M/M/m$

model with the same parameters.

Partitioning versus non-partitioning
When studying multiple-server polling models it is interesting to consider the option of partioning the system into a number of subsystems, each of which is served by one or more specific servers. Such a partitioning (or segmentation) seems to be particularly beneficial if the queues are clustered, and the switch-over times to move between the clusters are relatively large. On the other hand, when the system is partitioned into subsystems, the servers operating in mutually isolated clusters are not able to 'help' each other. Hence, it will happen from time to time that one server is idle while another server still has to serve a number of customers, so that the processing power is only partially used. As a consequence, in models with negligible switch-over times the amount of work in the system will always be smaller in the non-partitioned system.
To illustrate the effect of partitioning, consider the following model: $s = 4$; $m = 2$; $\boldsymbol{a} = (0.25, 0.25, 0.25, 0.25)$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00, 1.00)$; $m_i = 2$, $i = 1, \ldots, 4$; if $i, k \in \{1, 2\}$ or $i, k \in \{3, 4\}$ then $\sigma_{i,k}^{(1)} = 0.05$, otherwise $\sigma_{i,k}^{(1)} = \alpha$, $i, k = 1, \ldots, 4$. We compare the system performance between (i) the non-partitioned model, in which both servers visit each of the queues, with $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = (1, 2, 3, 4)$, and (ii) the partitioned model, in which server 1 serves Q_1 and Q_2 and server 2 serves Q_3 and Q_4, with $\boldsymbol{\pi}_1 = (1, 2)$ and $\boldsymbol{\pi}_2 = (3, 4)$. Table 6.4 gives the mean total amount of waiting work in the system (which is here proportional to the mean waiting time of an arbitrary customer) for various values of α and ρ, for the cases $\boldsymbol{q} = (0.00, 0.00, 0.00, 0.00)$ (i.e. 1-limited service at each queue) and $\boldsymbol{q} = (1.00, 1.00, 1.00, 1.00)$ (i.e. exhaustive service).

		$\boldsymbol{q} = (0.00, 0.00, 0.00, 0.00)$				$\boldsymbol{q} = (1.00, 1.00, 1.00, 1.00)$			
	$\rho \rightarrow$	0.4	0.8	1.6	1.8	0.4	0.8	1.6	1.8
α	0.01	0.10	0.28	2.30	6.48	0.10	0.27	1.96	4.63
$\downarrow$	0.05	0.14	0.34	2.72	9.14	0.14	0.31	2.08	4.87
	0.10	0.19	0.42	3.39	∞	0.18	0.37	2.23	5.17
	0.25	0.36	0.67	7.31	∞	0.33	0.56	2.69	6.11
	0.50	0.65	1.15	∞	∞	0.58	0.87	3.50	7.75
	1.00	1.27	2.43	∞	∞	1.08	1.50	5.24	11.93
partitioning		0.35	0.82	5.47	17.73	0.33	0.76	4.18	9.30

Table 6.4: Effect of partitioning.

Table 6.4 confirms the conjecture that partitioning will generally be disadvantageous when the switch-over times are relatively small. Moreover, it is illustrated that when the switch-over times become large, the loss of service capacity due to the switch-over times may tend to predominate the benefits from a non-partitioned system. Table 6.4 also suggests that the benefits of partitioning generally depend on the offered load to the system. In fact, when the

offered load is increased, it may well occur (for limited-type service disciplines) that some queues become instable in the non-partitioned system, whereas in the partitioned system all queues remain stable.

Comparisons with single-server models carrying a comparable load.
When investigating the performance of multiple-server polling models, it is interesting to make comparisons with single-server models with a comparable load. To this end, we first compare the performance of a multiple-server polling model (with m servers) with a single-server polling model in which the server operates at m-fold processing rate. Then, we will compare the multiple-server model with a single-server model with $1/m$-fold arrival rates.

Consider a multiple-server model versus a single-server model in which the server operates at m-fold processing rate. In the single-server case all processing power is concentrated, so that the single-server model might roughly be seen as a multiple-server model with extreme coalescence of the servers. Hence, one would expect that the waiting times at the queues are smaller in the multiple-server case, because the processing power would be more homogeneously distributed over the queues, cf. the discussion above. On the other hand, although the waiting times at the queues are expected to be smaller in the multiple-server case, the sojourn time (i.e. waiting time plus service time) of a customer in the system might be smaller in the single-server case (especially in light traffic), because the service times are (stochastically) smaller. In fact, for zero switch-over times the amount of work and hence, in a symmetrical model, the sojourn time, is smaller in the single-server m-speed case than in the multiple-server case.
As an illustration, we compare the system performance in both situations for the following model: $s = 4$; $\boldsymbol{a} = (0.25, 0.25, 0.25, 0.25)$; $\beta_i^{(1)} = 1/\mu$, $i = 1, \ldots, 4$; $m_i = 2$, $i = 1, \ldots, 4$; $\boldsymbol{q} = (1.00, 1.00, 1.00, 1.00)$ (i.e. exhaustive service); $\sigma_{i,k}^{(1)} = \alpha$, $i, k = 1, \ldots, 4$; $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = (1, 2, 3, 4)$. Table 6.5 shows the mean waiting time, $\mathrm{E}W$, of an arbitrary customer for the model with two servers both processing at normal speed ($\mu = 1.00$) and the model with a single server processing at double speed ($\mu = 2.00$). For the same models, Table 6.6 shows the mean sojourn times, defined by $\mathrm{E}R\,(= \mathrm{E}W + 1/\mu)$.

Tables 6.5 and 6.6 point out that the mean waiting times are generally smaller in the multiple-server case, but that this is not generally true for the mean sojourn times. When the system is operated by multiple normal-speed servers, instead of a single high-speed server, then (i) the mean waiting times decrease, and (ii) the mean service times increase. The results show that for small switch-over times the decrease of the mean service times dominates the increase of the mean waiting times, whereas the reverse is true when the switch-over times are relatively large.

We finally make a comparison between the performance of a single-server model and a multiple-server model (with m servers) with m-fold arrival rates. In gen-

		2 slow servers			1 fast server		
	$\rho \rightarrow$	0.4	1.6	1.8	0.4	1.6	1.8
α	0.01	0.06	1.84	4.38	0.15	2.09	4.66
$\downarrow$	0.10	0.23	2.37	5.45	0.41	2.85	6.10
	0.25	0.50	3.29	7.37	0.84	4.13	8.50
	0.50	0.95	4.96	10.95	1.56	6.25	12.50
	1.00	1.84	8.73	18.72	3.00	10.50	20.50

Table 6.5: Comparison: mean waiting times.

		2 slow servers			1 fast server		
	$\rho \rightarrow$	0.4	1.6	1.8	0.4	1.6	1.8
α	0.01	1.06	2.84	5.38	0.65	2.59	5.16
$\downarrow$	0.10	1.23	3.37	6.45	0.91	3.35	6.60
	0.25	1.50	4.29	8.37	1.34	4.63	9.00
	0.50	1.95	5.96	11.95	2.06	6.75	13.00
	1.00	2.84	9.73	19.72	3.50	11.00	21.00

Table 6.6: Comparison: mean sojourn times.

eral, multiple-server models tend to outperform single-server models with a proportional arrival stream, as the servers in a sense have the opportunity to cooperate. Stoyan [156] shows e.g. that in an ordinary $M/G/m$ model the mean waiting time is indeed always smaller than in an $M/G/1$ with proportional arrival rate. Although hard to prove, it is likely that in polling models the situation is similar. As an illustration, consider the model with the following parameters: $s = 4$; $\boldsymbol{a} = (0.25, 0.25, 0.25, 0.25)$; $\boldsymbol{\beta}^{(1)} = (1.00, 1.00, 1.00, 1.00)$; $m_i = 2$, $i = 1, \ldots, 4$; $q_i = 1.00$, $i = 1, \ldots, 4$ (i.e. exhaustive service); $\sigma_{i,k}^{(1)} = \alpha$, $i, k = 1, \ldots, 4$; $\boldsymbol{\pi}_1 = (1, 2, 3, 4)$. We have computed the mean waiting time for the single-server case and the two-server case (with $\boldsymbol{\pi}_2 = (1, 2, 3, 4)$) in which the arrival rate at each of the queues is doubled. Note that in the latter case the symmetry of the model implies that both servers carry the same load $\rho/2$. Table 6.7 shows the results for various values of α and offered load ρ.
Table 6.7 supports the conjecture that multiple-server models lead to a better system performance than single-server models with proportional arrival rates.

6.6 Approximation

In this section we propose a new mean waiting-time approximation for multiple-server polling models. It is assumed that each queue is served exhaustively, and that all server arrivals are effective. Moreover, each of the servers visits the

		single server				two servers			
	$\rho \rightarrow$	0.2	0.4	0.8	0.9	0.4	0.8	1.6	1.8
α	0.00	0.25	0.67	4.00	9.00	0.04	0.19	1.78	4.26
$\downarrow$	0.05	0.39	0.84	4.43	9.80	0.14	0.31	2.08	4.87
	0.25	0.97	1.54	6.13	13.00	0.50	0.77	3.29	7.37

Table 6.7: Comparison: mean waiting times.

queues in some cyclic order, which is not necessarily the same for all servers. The main motivation for proposing a new mean waiting-time approximation is that none of the existing waiting-time approximations (cf. [14], [99], [101], [111], [134], [139], [175]) takes into account the tendency of the servers to cluster when the system is heavily loaded, whereas this bunching effect (cf. Table 6.2) has been shown to have a considerable impact on the system performance (cf. Table 6.3). Hence, the accuracy of the existing approximation methods degrades significantly for heavily-loaded systems. The proposed approximation method *explicitly* takes into account the tendency of the servers to cluster. Numerical experiments have revealed that the proposed approximation yields accurate results. The proposed approximation method is only briefly sketched here. We refer to Borst and Van der Mei [36] and chapter 10 in Borst [32] for a more extensive discussion.

We first briefly review the single-server case. Let C_i be the cycle time of Q_i, defined as the time interval between two successive departures of the server from Q_i. For exhaustive service at Q_i, the mean waiting time at Q_i can be expressed in terms of the probability distribution of C_i as $EW_i = (1-\rho_i)E\tilde{C}_i$, where $E\tilde{C}_i = EC_i^2/2EC_i$ stands for the mean residual cycle time at Q_i (cf. [80], [93]). From simple balancing arguments it follows directly that $EC_i = \sigma_{1,1}/(1-\rho)$, $i = 1, \ldots, s$, so that it remains to approximate EC_i^2. The most common approach is to assume that the quantities $E\tilde{C}_i$ are equal for all i (cf. [80], [93]) and to substitute the so-obtained expression (with only one unknown) into the pseudo-conservation law (PCL), which is an exact expression for $\sum_{i=1}^{s} \rho_i EW_i$ (cf. [38], [40]), yielding a simple closed-form approximation for EW_i, $i = 1, \ldots, s$.

For the multiple-server case we follow a similar approach as in the single-server case. Denote by p_i the (unknown) probability that at least one of the servers is busy at Q_i, $i = 1, \ldots, s$. Define $f_{i,j}$ as the probability that exactly j servers are simultaneously working at Q_i, $i = 1, \ldots, s$, $j = 0, 1, \ldots, m$. Then the average number of servers working at Q_i simultaneously, given at least one server is busy at that queue, is equal to $\alpha_i := \Sigma_{j=1}^{m} j f_{i,j} / \Sigma_{j=1}^{m} f_{i,j} = \rho_i / p_i$, where the equality $\Sigma_{j=1}^{m} j f_{i,j} = \rho_i$ follows from simple balancing arguments. To derive an approximate relationship of the form $EW_i \approx \gamma_i E\tilde{C}_i$, we assume that the cus-

tomers experience the presence of multiple servers *as if* there is a single server processing at the same speed as $\alpha_i = \rho_i/p_i$ servers together. Similar arguments as for the single-server case lead to the following expression (cf. [36], [32]): for $i = 1, \ldots, s$,

$$\mathrm{E}W_i \approx (1 - p_i)\mathrm{E}\tilde{D}_i, \tag{6.25}$$

where $\mathrm{E}\tilde{D}_i = \mathrm{E}D_i^2/2\mathrm{E}D_i$, with D_i the time between two successive server departures from Q_i. However, as the degree of clustering may differ significantly from queue to queue, it is no longer reasonable to assume that the residual server interdeparture times, $\mathrm{E}\tilde{D}_i$, are approximately equal. Instead, $\alpha_i = \rho_i/p_i$, i.e. the average number of busy servers at Q_i, given Q_i is being served by at least one server, seems to give an indication for the residual server interdeparture times. We assume that residual server interdeparture times are proportional to the average processing speed which is used as a measure for the degree of clustering at Q_i, i.e. for $i = 1, \ldots, s$,

$$\mathrm{E}\tilde{D}_i \approx x\rho_i/p_i, \tag{6.26}$$

where x is some unknown constant. So it remains to find an expression for the unknown quantity x and the probabilities p_i, $i = 1, \ldots, s$.

To find approximative expressions for x, we follow a similar PCL-based approach as in the single-server case, as sketched above. However, in the multiple-server case the well-known work conservation property is no longer generally valid, let alone the property of work decomposition, which is in fact the basis for the PCLs. Therefore, we derive approximate PCLs to find an expression for $\sum_{i=1}^{s} \rho_i \mathrm{E}W_i$ and hence, for the unknown quantity x. The reader is referred to [36] for a more detailed derivation.

To approximate p_i, the probability that at least one of the servers is busy at Q_i, it should be noted that p_i can be expressed in terms of the distribution of the supplementary variables $(\boldsymbol{H}, \boldsymbol{Z})$ as follows (cf. section 6.3.1): for $i = 1, \ldots, s$,

$$p_i = 1 - \Pr\{(\pi_j(\boldsymbol{H}_j, \boldsymbol{Z}_j) \neq (i, 1),\ j = 1, \ldots, m\}. \tag{6.27}$$

It is clear that the process $\{(\boldsymbol{H}(t), \boldsymbol{Z}(t),\ t \geq 0\}$ is not a Markov process, because the transition rates will generally depend on the status of the queue-length vector $\boldsymbol{N}(t)$. However, to approximate the distribution of $(\boldsymbol{H}, \boldsymbol{Z})$, we will deal with the process $\{(\boldsymbol{H}(t), \boldsymbol{Z}(t),\ t \geq 0\}$ *as if* it is a Markov process, i.e. as if the transitions occur at a constant rate. These rates, in turn, can be specified as the reciprocal of the approximated means of the respective inter-transition times. To capture the phenomenon of clustering of the servers, we slightly *modify* the transitions for the states in which more than one server is busy at the same queue simultaneously. For these states we replace the transitions where *one* server leaves the queue by a single transition of the same rate where *all* the visiting servers leave the queue simultaneously, reflecting that

actually all the servers will tend to leave that queue relatively shortly after one another.
Now that all transition rates have been specified, the probability distribution of $(\boldsymbol{H}, \boldsymbol{Z})$ can be calculated by solving the balance equations, supplemented with the normalization condition. From that distribution the probabilities p_i, $i = 1, \ldots, s$, can be approximated according to (6.27).
Numerous numerical experiments have been done to compare the approximated values with the 'exact' values obtained with the PSA. These experiments have indicated that the approximation yields fairly accurate results for a wide variety of combinations of model parameters, even when the tendency of the servers to cluster is significant. The reader is referred to [31], [36] for a more detailed discussion of the various aspects of the proposed approximation method and for an extensive overview of the numerical results.

6.7 Concluding remarks

We have considered a polling model with multiple uncoupled servers in which each server visits the queues according to a given order. In general, such models are mathematically intractable. In this chapter it is shown how the model can be implemented into the PSA, a device for the numerical evaluation (and optimization) of performance measures of the system. This implementation into the PSA has been used for various numerical experiments. We have observed some interesting phenomena, such as the tendency of the servers to bunch together (cf. also [134]), and the fact that this coalescence of the servers deteriorates the system performance. We have also observed that the service orders may have a considerable impact on the system performance, compared with single server models. We have considered the option of partitioning the system into a number of subsystems. The latter has been shown to be particularly beneficial when the queues are somewhat clustered. Moreover, we have made comparisons of the performance of m-server models with the performance of (i) a system attended by a single server with m-fold processing rate and (ii) a system attended by a single server, where the arrival rates at the queues are divided by m. These comparisons have suggested that the mean waiting times (at the queues) are generally smaller in the multiple-server case. The time requirements of the PSA to compute performance measures of interest may be significant, so that there is a need to have sharp approximations for the waiting times at the queues. However, none of the existing waiting-time approximations takes into account the tendency of the servers to cluster when the system is highly loaded, whereas it is shown in section 6.5 that this clustering may have a considerable impact on the system performance. Therefore, we have proposed a new mean waiting-time approximation which explicitly takes into account the clustering effect. Numerical experiments have shown that the approximation method yields good results for a broad class of model parameters (cf. [36], [31]).

Finally, we discuss a number of topics for further research. For the case in

which each of the servers visits all queues in a strictly cyclic order (not necessarily identical for each server) and in which the servers incur the same switch-over times per cycle, consider the question: 'Does each server carry the same load?'. In the papers that have appeared in the literature it is assumed that the servers indeed carry the same load. Based on simulation results, Morris and Wang [134] did not find any significant differences between the loads carried by each of the servers. We did not find significant differences with the aid of the PSA either. This interesting phenomenon might be explained by the following intuitive argument. Consider a model with two servers. Assume that the mean switch-over time incurred per cycle is $\sigma_{1,1}$ for both servers. Then from simple balancing arguments it follows that the respective mean cycle times are given by $\mathrm{E}C_j = \sigma_{1,1}/(1-\tau_j)$, where τ_j is the load carried by server j, $j = 1, 2$. Suppose $\mathrm{E}C_1 < \mathrm{E}C_2$. Then, because of the fact that server 1 is moving around faster, server 1 will visit the queues more frequently and is likely to find more customers to be served. The latter would imply $\tau_1 > \tau_2$, which is a contradiction. We emphasize that this argument is only intuitive. Therefore, it would be interesting to investigate this intriguing question further.

Another interesting point is the following. For the single-server case with cyclic server routing, Levy et al. [123] proved that the amount of work in the system (at any time) is minimal when all queues are served exhaustively, i.e. when the server only leaves a queue when there are no waiting customers left at that queue. One may construct examples showing that for multiple-server polling the latter is not the case in a sample-path sense. Nevertheless, this does not exclude that the monotonicity with respect to the 'exhaustiveness' of the service disciplines may hold for the steady-state amount of work in the system. However, it is not impossible that in the multiple-server case the steady-state system performance may be improved by serving non-exhaustively, because the bunching effect (discussed earlier) would be alleviated. It would be interesting to investigate this monotonicity further.

Moreover, one may consider models in which the service discipline of a server arriving at a queue may depend on the number of servers already working at that queue. In such a model, an arriving server might be forced to skip a queue where a relatively large number of servers is already working. In this way, the bunching effect would be alleviated. It would be interesting to investigate the performance of this type of models in more detail.

In the model description in section 6.2 the number of servers working at Q_i simultaneously may be restricted by a maximum m_i. Numerical experiments have suggested that decreasing the values of m_i deteriorates the system performance, which seems to be intuitively clear. However, in heavily-loaded systems with negligible switch-over times, tightening the restrictions on the number of servers simultaneously working at the queues may somewhat disperse the server-position distribution and possibly alleviate the coalescing effect, leading to a better system performance. It would be an interesting topic for further re-

search to investigate whether putting such an extra restriction on the behavior of the servers may improve the system performance.

Bibliography

[1] M. Ajmone Marsan, L.F. De Moraes, S. Donatelli and F. Neri (1990). Analysis of symmetric nonexhaustive polling with multiple servers. In: *Proc. INFOCOM '90*, 284–295.

[2] M. Ajmone Marsan, L.F. De Moraes, S. Donatelli and F. Neri (1992). Cycles and waiting times in symmetric exhaustive and gated multiserver multiqueue systems. In: *Proc. INFOCOM '92*, 2315–2324.

[3] M. Ajmone Marsan, S. Donatelli and F. Neri (1990). GSPN models of Markovian multiserver multiqueue systems. *Perf. Eval.* **11**, 227–240.

[4] M. Ajmone Marsan, S. Donatelli and F. Neri (1991). Multiserver multiqueue systems with limited service and zero walk time. In: *Proc. INFOCOM '91*, 1178–1188.

[5] E. Altman (1994). Analysing timed-token ring protocols using the power-series algorithm. In: *The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks*, eds. J. Labetoulle and J.W. Roberts (Elsevier, Amsterdam), 961–971.

[6] E. Altman, J.P.C. Blanc, A. Khamishy and U. Yechiali (1992). Polling systems with walking and switch-in times. Technical Report INRIA, Sophia Antipolis, France.

[7] E. Altman, P. Konstantopoulos and Z. Liu (1992). Some qualitative properties for polling systems. *Queueing Systems* **11**, 35–57.

[8] E. Altman and U. Yechiali (1994). Polling in a closed network. To appear in *Prob. Eng. Inf. Sc.* **8**.

[9] S. Asmussen (1987). *Applied Probability and Queues* (Wiley, Chichester).

[10] K.B. Athreya and P.E. Ney (1972). *Branching Processes* (Springer-Verlag, Berlin).

[11] G.A. Baker and P. Graves-Morris (1981). *Padé Approximants, Part I: Basic Theory* (Addison-Wesley Publ. Cy., Massachusetts, 2nd ed.).

[12] G.A. Baker and P. Graves-Morris (1981). *Padé Approximants, Part II: Extensions and Applications* (Addison-Wesley Publ. Cy., Massachusetts, 2nd ed.).

[13] J.E. Baker and I. Rubin (1987). Polling with a general service order table. *IEEE Trans. Commun.* **35**, 283–288.

[14] L.N. Bhuyan, D. Ghosal and Q. Yang (1989). Approximate analysis of single and multiple ring networks. *IEEE Trans. Comput.* **38**, 1027–1040.

[15] J.P.C. Blanc (1987). A note on waiting times in systems with queues in parallel. *J. Appl. Prob.* **24**, 540–546.

[16] J.P.C. Blanc (1987). On a numerical method for calculating state-probabilities for queueing systems with more than one waiting line. *J. Comp. Appl. Math.* **20**, 119–125.

[17] J.P.C. Blanc (1988). A numerical study of a coupled-processor model. In: *Computer Performance and Reliability*, eds. G. Iazeolla, P.J. Courtois and O.J. Boxma (North-Holland, Amsterdam), 289–303.

[18] J.P.C. Blanc (1990). A numerical approach to cyclic-service queueing models. *Queueing Systems* **6**, 173–188.

[19] J.P.C. Blanc (1990). Cyclic polling systems: limited versus Bernoulli service. Report FEW 422, Tilburg University, The Netherlands.

[20] J.P.C. Blanc (1991). The power-series algorithm applied to cyclic polling systems. *Stoch. Mod.* **7**, 527–545.

[21] J.P.C. Blanc (1992). An algorithmic solution of polling systems with limited service disciplines. *IEEE Trans. Commun.* **40**, 1152–1155.

[22] J.P.C. Blanc (1992). Performance evaluation of polling systems by means of the power-series algorithm. *Ann. Oper. Res.* **35**, 155–186.

[23] J.P.C. Blanc (1992). The power-series algorithm applied to the shortest-queue model. *Oper. Res.* **40**, 157–167.

[24] J.P.C. Blanc (1993). Performance analysis and optimization with the power-series algorithm. In: *Performance Evaluation of Computer and Communication Systems*, eds. L. Donatiello and R. Nelson (North-Holland, Amsterdam), 53–80.

[25] J.P.C. Blanc and R.D. van der Mei (1992). Optimization of polling systems by means of gradient methods and the power-series algorithm. Report FEW 575, Tilburg University, The Netherlands.

[26] J.P.C. Blanc and R.D. van der Mei (1994). Computation of derivatives by means of the power-series algorithm. To appear in *ORSA J. Comput.*

[27] J.P.C. Blanc and R.D. van der Mei (1994). The power-series algorithm applied to polling systems with a dormant server. In: *The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks*, eds. J. Labetoulle and J.W. Roberts (Elsevier, Amsterdam), 865–874.

[28] J.P.C. Blanc and R.D. van der Mei (1995). Optimization of polling systems with Bernoulli schedules. *Perf. Eval.* **2**, 139–158.

[29] S.C. Borst (1993). A polling system with a homing server. CWI Report BS-R9313, Amsterdam.

[30] S.C. Borst (1994). A pseudo-conservation law for a polling system with a dormant server. In: *The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks*, eds. J. Labetoulle and J.W. Roberts (Elsevier, Amsterdam), 729–742.

[31] S.C. Borst (1994). *Polling Systems.* Ph.D. Thesis, Tilburg University, The Netherlands.

[32] S.C. Borst (1994). Polling systems with multiple coupled servers. CWI Report BS-R9408, Amsterdam.

[33] S.C. Borst and O.J. Boxma (1994). Polling models with and without switchover times. CWI Report BS-R9421, Amsterdam.

[34] S.C. Borst, O.J. Boxma, J.H.A. Harink and G.B. Huitema (1994). Optimization of fixed time polling schemes. *Telecommun. Syst.* **3**, 31–59.

[35] S.C. Borst, O.J. Boxma and H. Levy (1993). The use of service limits for efficient operation of multi-station single-medium communication systems. CWI Report BS-R9312, Amsterdam.

[36] S.C. Borst and R.D. van der Mei (1994). Waiting-time approximations for multiple-server polling systems. CWI Report BS-R9428, Amsterdam. Submitted.

[37] O.J. Boxma (1986). Models of two queues: a few new views. In: *Teletraffic Analysis and Computer Performance Evaluation*, eds. O.J. Boxma, J.W. Cohen and H.C. Tijms (North-Holland, Amsterdam), 75–98.

[38] O.J. Boxma (1989). Workloads and waiting times in single-server systems with multiple customer classes. *Queueing Systems* **5**, 185–214.

[39] O.J. Boxma (1991). Analysis and optimization of polling systems. In: *Queueing, Performance and Control in ATM*, eds. J.W. Cohen and C.D. Pack (North-Holland, Amsterdam), 173–183.

[40] O.J. Boxma and W.P. Groenendijk (1987). Pseudo-conservation laws in cyclic-service systems. *J. Appl. Prob.* **24**, 949–964.

[41] O.J. Boxma and W.P. Groenendijk (1988). Two queues with alternating service and switching times. In: *Queueing Theory and its Applications Liber Amicorum for J.W. Cohen*, eds. O.J. Boxma and R. Syski (North-Holland, Amsterdam), 261-282.

[42] O.J. Boxma and W.P. Groenendijk (1988). Waiting times in discrete-time cyclic-service systems. *IEEE Trans. Commun.* **36**, 164–170.

[43] O.J. Boxma, W.P. Groenendijk and J.A. Weststrate (1990). A pseudo-conservation law for service systems with a polling table. *IEEE Trans. Commun.* **38**, 1865–1870.

[44] O.J. Boxma, H. Levy and J.A. Weststrate (1991). Efficient visit frequencies for polling tables: minimization of the waiting cost. *Queueing Systems* **9**, 133–162.

[45] O.J. Boxma, H. Levy and J.A. Weststrate (1993). Efficient visit orders for polling systems. *Perf. Eval.* **18**, 103–123.

[46] O.J. Boxma, H. Levy and U. Yechiali (1992). Cyclic reservation schemes for efficient operation of multiple-queue single-server systems. *Ann. Oper. Res.* **35**, 187–208.

[47] O.J. Boxma and B. Meister (1987). Waiting-time approximations for cyclic-service systems with switch-over times. *Perf. Eval.* **7**, 299–308.

[48] O.J. Boxma and J.A. Weststrate (1989). Waiting times in polling systems with Markovian server routing. In: *Messung, Modellierung und Bewertung von Rechensystemen und Netze*, eds. G Stiege and J.S. Lie (Springer, Berlin), 89–104.

[49] Y.A. Bozer and M.M. Srinivasan (1991). Tandem configurations for automated guided vehicle systems and the analysis of single loops. *IIE Trans.* **23**, 72–82.

[50] C. Brezinsky (1980). *Padé-type Approximation and General Orthogonal Polynomials* (Birkhäuser, Basel).

[51] S. Browne, E.G. Coffman jr., E.N. Gilbert and P.E.W. Wright (1992). Gated, exhaustive, parallel service. *Prob. Eng. Inf. Sc.* **6**, 217–239.

[52] S. Browne and O. Kella (1992). Parallel service with vacations. Technical Report Columbia University, New York, NY.

[53] S. Browne and G. Weiss (1992). Dynamic priority rules when polling with multiple parallel servers. *Oper. Res. Lett.* **12**, 129–137.

[54] S. Browne and U. Yechiali (1989). Dynamic priority rules for cyclic-type queues. *Adv. Appl. Prob.* **21**, 432–450.

[55] S.L. Brumelle (1971). On the relation between customer and time averages in queues. *J. Appl. Prob.* **8**, 508–520.

[56] B.D. Bunday and W.K. El-Badri (1988). The efficiency of M groups of machines served by a travelling robot: comparison of two models. *Int. J. Prod. Res.* **26**, 299–308.

[57] W. Bux (1989). Token-ring local area networks and their performance. *Proc. IEEE* **77**, 238–256.

[58] C. Buyukkoc, P. Varaiya and J. Walrand (1985). The $c\mu$ rule revisited. *Adv. Appl. Prob.* **17**, 237–238.

[59] K.C. Chang and D. Shandu (1992). Mean waiting time approximations in cyclic-service systems with exhaustive-limited service policy. *Perf. Eval.* **15**, 21–40.

[60] C. Chatfield and A.J. Collins (1980). *Introduction to Multivariate Analysis* (Chapman and Hall, London).

[61] G.L. Choudhury (1990). Polling with a general service order table: gated service. In: *Proc. INFOCOM '90*, 268–276.

[62] G.L. Choudhury and W. Whitt (1994). Computing transient and steady-state distributions in polling models by numerical transform inversion. To appear in *Perf. Eval.*

[63] H. Chung, C.K. Un and W.Y. Jung (1994). Performance analysis of Markovian polling systems with single buffers. *Perf. Eval.* **19**, 303–315.

[64] E.G. Coffman Jr., G. Fayolle and I. Mitrani (1988). Two queues with alternating service periods. In: *Proc. Performance '87*, eds. P.J. Courtois and G. Latouche (North-Holland, Amsterdam), 227–239.

[65] J.W. Cohen (1982). *The Single Server Queue* (North-Holland, Amsterdam, 2nd ed.).

[66] J.W. Cohen (1988). A two-queue model with semi-exhaustive alternating service. In: *Proc. Performance '87*, eds. P.J. Courtois and G. Latouche (North-Holland, Amsterdam), 19–37.

[67] J.W. Cohen and O.J. Boxma (1981). The M/G/1 queue with alternating service formulated as a Riemann-Hilbert boundary value problem. In: *Proc. Performance '81*, ed. F.J. Kylstra (North-Holland, Amsterdam), 181–199.

[68] J.W. Cohen and O.J. Boxma (1983). *Boundary Value Problems in Queueing System Analysis* (North-Holland, Amsterdam).

[69] M. Conti, E. Gregori and L. Lenzini (1993). Metropolitan area networks (MANs): protocols, modeling and performance evaluation. In: *Performance Evaluation of Computer and Communication Systems*, eds. L. Donatiello and R. Nelson (North-Holland, Amsterdam), 81–120.

[70] R.B. Cooper (1970). Queues served in cyclic order: waiting times. *Bell. Syst. Techn. J.* **49**, 399–413.

[71] R.B. Cooper and G. Murray (1969). Queues served in cyclic order. *Bell. Syst. Techn. J.* **48**, 675–689.

[72] R.B. Cooper, S.C. Niu and M.M. Srinivasan (1992). A decomposition theorem for polling models: the switchover times are effectively additive. To appear in *Oper. Res.*

[73] E. De Souza e Silva, H.R. Gail and R.R. Muntz (1993). Polling systems with server timeouts. Preprint.

[74] C.F. Daganzo (1988). Some properties of polling systems. Report Department of Civil Engineering, UCLA.

[75] I.M. Dukhovnyy (1979). An approximate model of motion of urban passenger transportation over annular routes. *Eng. Cybern.* **17**, 161–162.

[76] M. Eisenberg (1971). Two queues with changeover times. *Oper. Res.* **19**, 386–401.

[77] M. Eisenberg (1972). Queues with periodic service and changeover time. *Oper. Res.* **20**, 440–451.

[78] M. Eisenberg (1979). Two queues with alternating service. *SIAM J. Appl. Math.* **36**, 287–303.

[79] M. Eisenberg (1994). The polling system with a stopping server. *Queueing Systems* **18**, 387–431.

[80] D. Everitt (1986). Simple approximations for token rings. *IEEE Trans. Commun.* **34**, 719–721.

[81] O. Fabian and H. Levy (1994). Pseudo-cyclic policies for multi-queue single-server systems. *Ann. Oper. Res.* **48**, 127–152.

[82] A. Federgruen and Z. Katalan (1994). Approximating queue size and waiting time distributions in general polling systems. *Queueing Systems* **18**, 353–386.

[83] M.J. Ferguson and Y.J. Aminetzah (1985). Exact results for nonsymmetric token ring systems. *IEEE Trans. Commun.* **33**, 223–231.

[84] L. Fournier and Z. Rosberg (1991). Expected waiting times in cyclic service systems under priority disciplines. *Queueing Systems* **9**, 419–439.

[85] C. Fricker and M.R. Jaïbi (1994). Monotonicity and stability of periodic polling models. *Queueing Systems* **15**, 211–238.

[86] C. Fricker and M.R. Jaïbi (1994). Stability of a polling model with a Markovian scheme. Technical Report INRIA 2278, France.

[87] S.W. Fuhrmann (1992). A decomposition result for a class of polling models. *Queueing Systems* **11**, 109–120.

[88] S.W. Fuhrmann and Y.T. Wang (1988). Analysis of cyclic service systems with limited service: bounds and approximations. *Perf. Eval.* **9**, 35–54.

[89] B. Gamse and G.F. Newell (1982). An analysis of elevator operation in moderate height buildings - I. A single elevator. *Transp. Res. B* **16**, 303–319.

[90] B. Gamse and G.F. Newell (1982). An analysis of elevator operation in moderate height buildings - II. Multiple elevators. *Transp. Res. B* **16**, 321-335.

[91] P.E. Gill, W. Murray and M.H. Wright (1981). *Practical Optimization* (Academic Press, New York).

[92] D. Grillo (1990). Polling mechanism models in communication systems - some application examples. In: *Stochastic Analysis of Computer and Communication Systems*, ed. H. Takagi (North-Holland, Amsterdam), 659–699.

[93] W.P. Groenendijk (1989). Waiting-time approximations for cyclic-service systems with mixed service strategies. In: *Teletraffic Science for New Cost-Effective Systems*, ed. M. Bonatti (North-Holland, Amsterdam), 1434–1441.

[94] W.P. Groenendijk (1990). *Conservation Laws in Polling Systems.* Ph.D. Thesis, University of Utrecht, The Netherlands.

[95] D. Gupta and M.M. Srinivasan (1993). When should a roving server be patient? Technical Report University of Tennessee.

[96] M. Hofri and K.W. Ross. On the optimal control of two queues with server setup times and its analysis. *SIAM J. Comput.* **16**, 399–420.

[97] G. Hooghiemstra, M.S. Keane and S. van de Ree (1988). Power series for stationary distributions of coupled processor models. *SIAM J. Appl. Math.* **48**, 1159–1166.

[98] A. Itai and Z. Rosberg (1984). A Golden Ratio control policy for a multiple-access channel. *IEEE Trans. Autom. Contr.* **29**, 712–718.

[99] A.E. Kamal and V.C. Hamacher (1989). Approximate analysis of non-exhaustive multi-server polling systems with applications to local area networks. *Comput. Netw. ISDN Syst.* **17**, 15–27.

[100] E.P.C. Kao and K.S. Narayanan (1991). Analyses of an M/M/N queue with servers' vacations. *EJOR* **54**, 256–266.

[101] V.V. Karmarkar and J.G. Kuhl (1989). An integrated approach to distributed demand assignment in multiple-bus local networks. *IEEE Trans. Comput.* **38**, 679–695.

[102] J. Keilson and L.D. Servi (1986). Oscillating random walk models for GI/G/1 vacation systems with Bernoulli schedules. *J. Appl. Prob.* **23**, 790–802.

[103] W.B. Kim and E. Koenigsberg (1987). The efficiency of two groups of N machines served by a single robot. *J. Oper. Res. Soc.* **38**, 523–538.

[104] L. Kleinrock and H. Levy (1988). The analysis of random polling systems. *Oper. Res.* **36**, 716–732.

[105] E. Koenigsberg and J. Mamer (1982). The analysis of production systems. *Int. J. Prod. Res.* **20**, 1–16.

[106] A.G. Konheim and H. Levy (1992). Efficient analysis of polling systems. In: *Proc. INFOCOM '92*, 2325–2331.

[107] A.G. Konheim, H. Levy and M.M. Srinivasan (1994). Descendant set: an efficient approach for the analysis of polling systems. *IEEE Trans. Commun.* **42**, 1245–1253.

[108] G. Koole (1994). Assigning a single server to inhomogeneous queues with switching costs. CWI Report BS-R9405, Amsterdam.

[109] G. Koole (1994). On the power-series algorithm. In: *Evaluation of Parallel and Distributed Systems-Solution Methods*, eds. O.J. Boxma and G. Koole, CWI Tract 105 & 106 (CWI, Amsterdam), 139–155.

[110] J.B. Kruskal (1969). Work-scheduling algorithms: a non-probabilistic queueing study (with possible applications to No. 1 ESS). *Bell Syst. Techn. J.* **48**, 2963–2974.

[111] A.C. Lavelha, J. Moreira de Souza and J.B. Ribeiro do Val (1994). Approximate analysis of multiqueue systems with multiple cyclic queues. *Perf. Eval.* **20**, 391-412.

[112] D.S. Lee (1994). Analysis of a cyclic server queue with Bernoulli service. C&C Research Laboratories, N.E.C. U.S.A., Princeton, NJ.

[113] D.S. Lee and B. Sengupta (1994). Analysis of a cyclic server queue with the exhaustive and limited service discipline. C&C Research Laboratories, N.E.C. U.S.A., Princeton, NJ.

[114] K.K. Leung (1991). Cyclic-service systems with probabilistically-limited service. *IEEE J. Sel. Areas Commun.* **9**, 185–193.

[115] K.K. Leung (1994). Cyclic-service systems with non-preemptive, time-limited service. *IEEE Trans. Commun.* **42**, 2521–2524.

[116] H. Levy (1984). *Non-Uniform Structures and Synchronization Patterns in Shared-Channel Communication Networks.* Ph.D. Thesis, UCLA.

[117] H. Levy (1988). Optimization of polling systems: the fractional exhaustive method. Technical Report Tel-Aviv University.

[118] H. Levy (1989). Analysis of cyclic polling systems with binomial-gated service. In: *Performance of Distributed and Parallel Systems*, eds. T. Hasegawa, H. Takagi and Y. Takahashi (North-Holland, Amsterdam), 127–139.

[119] H. Levy and L. Kleinrock (1991). Polling systems with zero switch-over periods: a general method for analyzing the expected delay. *Perf. Eval.* **13**, 97–107.

[120] H. Levy, G. Mahalal and M. Sidi (1994). Multi-token rings and multi-server polling systems: the bang-bang policy. Preprint.

[121] H. Levy and M. Sidi (1990). Polling systems: applications, modeling and optimization. *IEEE Trans. Commun.* **38**, 1750–1760.

[122] H. Levy and M. Sidi (1991). Polling systems with simultaneous arrivals. *IEEE Trans. Commun.* **39**, 823-827.

[123] H. Levy, M. Sidi and O.J. Boxma (1990). Dominance relations in polling systems. *Queueing Systems* **6**, 155–171.

[124] Y. Levy and U. Yechiali (1976). An M/M/s queue with servers' vacations. *INFOR* **14**, 153–163.

[125] Z. Liu and P. Nain (1992). Optimal scheduling in some multi-queue single-server systems. *IEEE Trans. Autom. Contr.* **37**, 247–252.

[126] Z. Liu, P. Nain and D. Towsley (1992). On optimal polling policies. *Queueing Systems* **11**, 59–83.

[127] W.M. Loucks, V.C. Hamacher, B.R. Preiss and L. Wong (1985). Short-packet transfer performance in local area ring networks. *IEEE Trans. Comput.* **34**, 1006–1014.

[128] D.M. Lucantoni (1991). New results on the single server queue with batch Markovian arrival processes. *Stoch. Mod.* **7**, 1–46.

[129] C. Mack, T. Murphy and N.L. Webb (1957). The efficiency of N machines unidirectionally patrolled by one operative when walking times and repair times are constants. *J. Royal Stat. Soc.* B **19**, 166-172.

[130] I. Meilijson and U. Yechiali (1977). On optimal right-of-way policies at a single-server station where insertion of idle times is permitted. *Stoch. Proc. Appl.* **6**, 23–52.

[131] I. Mitrani (1982). *Simulation Techniques for Discrete Event Systems* (Cambridge University Press, Cambridge).

[132] I. Mitrani, J.L. Adams and R.M. Falconer (1986). A modelling study of the Orwell ring protocol. In: *Teletraffic Analysis and Computer Performance Evaluation*, eds. O.J. Boxma, J.W. Cohen and H.C. Tijms (North-Holland, Amsterdam), 429–438.

[133] I.L. Mitrany and B. Avi-Itzhak (1968). A many-server queue with server interruptions. *Oper. Res.* **16**, 628–638.

[134] R.J.T. Morris and Y.T. Wang (1984). Some results for multi-queue systems with multiple cyclic servers. In: *Performance of Computer-Communications Systems*, eds. W. Bux and H. Rudin (North-Holland, Amsterdam), 245–258.

[135] S. Nahmias and M.H. Rothkopf (1984). Stochastic models for internal mail delivery systems. *Mgmt. Sc.* **30**, 1113–1120.

[136] M.F. Neuts (1981). *Matrix-Geometric Solutions in Stochastic Models: an Algorithmic Approach* (Johns Hopkins University Press, Baltimore).

[137] M.F. Neuts and D.M. Lucantoni (1979). A Markovian queue with N servers subject to breakdowns and repairs. *Mgmt. Sc.* **25**, 849–861.

[138] G.F. Newell (1969). Properties of vehicle-actuated signals: I One-way streets. *Transp. Sc.* **3**, 99–125.

[139] T. Raith (1985). Performance analysis of multibus interconnection networks in distributed systems. In: *Teletraffic Issues in an Advanced Information Society*, ed. M. Akiyama (North-Holland, Amsterdam), 662–668.

[140] R. Ramaswamy and L.D. Servi (1988). The busy period of the M/G/1 vacation model with Bernoulli schedules. *Stoch. Mod.* **4**, 507–521.

[141] S.S. Rao (1984). *Optimization Theory and Applications* (Wiley Eastern Limited, New Delhi, 2nd ed.).

[142] J.A.C. Resing (1993). Polling systems and multi-type branching processes. *Queueing Systems* **13**, 409–426.

[143] R.Y. Rubinstein and A. Shapiro (1993). *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization via the Score Function Method* (John Wiley & Sons, New York).

[144] D. Sarkar and W.I. Zangwill (1988). Variance effects in cyclic production systems. *Mgmt. Sc.* **34**, 444–453.

[145] D. Sarkar and W.I. Zangwill (1989). Expected waiting time for nonsymmetric cyclic queueing systems - Exact results and applications. *Mgmt. Sc.* **35**, 1463–1474.

[146] C.H. Sauer and E.A. MacNair (1983). *Simulation of Computer Communication Systems* (Prentice Hall, Englewood Cliffs).

[147] R. Schassberger (1973). *Warteschlangen* (Springer-Verlag, Berlin).

[148] E. Seneta (1981). *Non-Negative Matrices and Markov Chains* (Springer, New York, 2nd ed.).

[149] L.D. Servi (1986). Average delay approximation of M/G/1 cyclic service queues with Bernoulli schedules. *IEEE J. Sel. Areas Commun.* **4**, 813–822.

[150] S. Shimogawa and Y. Takahashi (1992). A note on the conservation law for a multi-queue with local priority. *Queueing Systems* **11**, 145–151.

[151] M. Sidi and H. Levy (1990). Customer routing in polling systems. In: *Proc. Performance '90*, eds. P.J.B. King, I. Mitrani and R.B. Pooley (North-Holland, Amsterdam), 319–331.

[152] M.M. Srinivasan (1988). An approximation for mean waiting times in cyclic server systems with non-exhaustive service. *Perf. Eval.* **9**, 17–33.

[153] M.M. Srinivasan (1988). Non-deterministic polling systems. *Mgmt. Sc.* **37**, 667–681.

[154] M.M. Srinivasan, H. Levy and A.G. Konheim (1993). The individual station technique for the analysis of cyclic polling systems. Technical Report University of Tennessee.

[155] M.M. Srinivasan, S.C. Niu and R.B. Cooper (1993). Relating polling models with nonzero and zero switchover times. Technical Report University of Tennessee. To appear in *Queueing Systems.*

[156] D. Stoyan (1974). Some bounds for many-server systems GI/G/s. *Math. Oper. Stat.* **5**, 117–129.

[157] L. Takács (1968). Two queues attended by a single server. *Oper. Res.* **16**, 639–650.

[158] H. Takagi (1986). *Analysis of Polling Systems* (The MIT Press, Cambridge, Massachusetts).

[159] H. Takagi (1990). Queueing analysis of polling models: an update. In: *Stochastic Analysis of Computer and Communication Systems*, ed. H. Takagi (North-Holland, Amsterdam), 267–318.

[160] H. Takagi (1991). Applications of polling models to computer networks. *Comp. Netw. ISDN Syst.* **22**, 193–211.

[161] H. Takagi (1994). Queueing analysis of polling models: progress in 1990-1993. To appear in: *Frontiers in Queueing: Models, Methods and Problems*, ed. J.H. Dsalalow (CRC Press).

[162] T.E. Tedijanto (1990). Exact results for the cyclic-service queue with a Bernoulli schedule. *Perf. Eval.* **11**, 107–115.

[163] T.E. Tedijanto (1990). *Nonexhaustive Policies in Polling Systems and Vacation Models.* Ph.D. Thesis, University of Maryland.

[164] T.E. Tedijanto (1992). A note on the comparison between Bernoulli and limited policies in vacation models. *Perf. Eval.* **15**, 89–97.

[165] B. van Arem (1990). *Queueing Network Models for Slotted Transmission Systems.* Ph.D. Thesis, Twente University, Enschede, The Netherlands.

[166] W.B. van den Hout and J.P.C. Blanc (1993). The power-series algorithm extended to the BMAP/PH/1 queue. To appear in *Stoch. Mod.*

[167] W.B. van den Hout and J.P.C. Blanc (1994). The power-series algorithm for a wide class of Markov processes. Center discussion paper 9487, Tilburg University, The Netherlands.

[168] W.B. van den Hout and J.P.C. Blanc (1994). The power-series algorithm for Markovian queueing networks. Center discussion paper 9467, Tilburg University, The Netherlands.

[169] R.D. van der Mei (1994). Polling systems with Markovian server routing. Center discussion paper 9514, Tilburg University, The Netherlands. Submitted.

[170] R.D. van der Mei and S.C. Borst (1994). Analysis of multiple-server polling systems by means of the power-series algorithm. CWI Report BS-R9410, Amsterdam. Submitted.

[171] J.A. Weststrate (1992). *Analysis and Optimization of Polling Systems.* Ph.D. Thesis, Tilburg University, The Netherlands.

[172] J.A. Weststrate and R.D. van der Mei (1994). Waiting times in a two-queue model with exhaustive and Bernoulli service. *Z.O.R. Mod. Meth. Oper. Res.* **40**, 289–303.

[173] R.W. Wolff (1982). Poisson arrivals see time averages. *Oper. Res.* **30**, 223–231.

[174] P. Wynn (1966). On the convergence and stability of the epsilon algorithm. *SIAM J. Num. Anal.* **3**, 91–122.

[175] Q. Yang, D. Ghosal and L.N. Bhuyan (1986). Performance analysis of multiple token rings and multiple slotted ring networks. In: *Proc. Comp. Netw. Symp.*, 79–86.

[176] U. Yechiali (1991). Optimal dynamic control of polling systems. In: *Queueing, Performance and Control in ATM*, eds. J.W. Cohen and C.D. Pack (North-Holland, Amsterdam), 195–208.

[177] M. Zafirovic-Vukotic, I.G. Niemegeers and D.S. Valk (1988). Performance modelling of slotted ring protocols in HSLAN's. *IEEE J. Sel. Areas Commun.* **6**, 1001–1024.

Samenvatting

Dit proefschrift is gewijd aan de analyse en optimalisatie van zogenaamde *polling*-modellen, een klasse van wachtrijmodellen. Een wachtrijmodel is een wiskundig model dat situaties beschrijft waarin verschillende gebruikers ('klanten') diensten vragen van één of andere faciliteit ('bediende'). Een gebruiker die niet direct bediend kan worden, neemt plaats in een wachtrij. Het klassieke polling-model bestaat uit een aantal wachtrijen en één bediende, die de rijen in een bepaalde volgorde bezoekt om aldaar wachtende klanten te bedienen (zie Figuur 1.1). De *bedieningsdiscipline* bepaalt welke klanten tijdens een bezoek van de bediende aan een rij worden bediend. Het *routeringsmechanisme* bepaalt in welke volgorde de bediende de verschillende rijen bezoekt. Meestal wordt aangenomen dat het omschakelen van de bediende van de ene naar de andere wachtrij een zekere *omschakeltijd* vergt.
Polling-modellen worden gebruikt voor het analyseren van situaties waarin verschillende typen gebruiker bediening vragen van één gemeenschappelijke bedieningsfaciliteit. Zo kunnen polling-modellen bijvoorbeeld worden toegepast voor het analyseren van de prestatie van computersystemen, communicatienetwerken, verkeerssystemen, liftsystemen, productiesystemen en onderhoudsstrategieën.
De meeste polling-modellen zijn echter niet exact analyseerbaar met behulp van bestaande wiskundige technieken. Bovendien, zelfs al ís een polling-model formeel gezien exact analyseerbaar, dan nóg leidt de analyse niet altijd tot hanteerbare uitdrukkingen voor prestatiematen zoals gemiddelde wachttijden en rijlengten. Daardoor is de behoefte ontstaan aan numerieke technieken om dit soort prestatiematen te bepalen. Numerieke technieken leveren, in tegenstelling tot analytische methoden, geen exacte uitdrukkingen voor prestatiematen van het systeem, maar kunnen worden gebruikt om voor een *gegeven* model numerieke waarden (getallen) voor prestatiematen te bepalen. Eén van deze technieken is het zogenaamde *machtreeksalgoritme*. In dit proefschrift wordt het machtreeksalgoritme gebruikt voor het analyseren en optimaliseren van polling-modellen.

Overzicht van de hoofdstukken
In hoofdstuk 1 wordt een overzicht gegeven van verschillende aspecten van polling-modellen, zoals de toepassingsmogelijkheden, de verschillende modelvarianten en de stand van zaken op het gebied van de analyse en optimalisatie.

In hoofdstuk 2 wordt het *machtreeksalgoritme* besproken. In het eerste gedeelte wordt een overzicht gegeven van verschillende aspecten van het gebruik van het machtreeksalgoritme voor wachtrijmodellen met een zogenaamde multi-dimensionale geboorte-en-sterfte-structuur. In het tweede deel wordt beschreven hoe het machtreeksalgoritme ook kan worden aangewend voor het berekenen van *afgeleiden* van de prestatiematen met betrekking tot een algemene klasse van systeemparameters. Daardoor is het machtreeksalgoritme ook bruikbaar voor *optimalisatie* van de prestatie van het systeem met betrekking tot de systeemparameters.
De algemene beschrijving van het machtreeksalgoritme in dit hoofdstuk wordt gebruikt in de volgende hoofdstukken. Daarin wordt de prestatie van een aantal polling-modellen geanalyseerd en geoptimaliseerd met behulp van het machtreeksalgoritme.

In hoofdstuk 3 wordt ingegaan op het *optimaliseren* van polling-modellen met betrekking tot de *bedieningsdisciplines* bij de rijen. We beschouwen een polling-model waarin de bediende met een willekeurig aantal, zeg s, wachtrijen in een cyclische volgorde bezoekt en waarin de rijen worden bediend volgens de zogenaamde Bernoulli bedieningsdiscipline. Dat wil zeggen, als na de bediening van een klant in rij i nog andere klanten in die rij aanwezig zijn, dan wordt met kans q_i de volgende klant in rij i bediend, en met kans $1 - q_i$ gaat de bediende naar de volgende rij, $i = 1, \ldots, s$. We beschouwen het probleem van het bepalen van een combinatie van Bernoulli-parameters $\boldsymbol{q}^* = (q_1^*, \ldots, q_s^*)$ die een gegeven gewogen som van de verwachte wachttijden minimaliseert. In het algemeen is dit optimaliseringsprobleem niet exact oplosbaar. Aangetoond zal worden hoe het machtreeksalgoritme kan worden gebruikt om prestatiematen van het systeem en de afgeleides naar de Bernoulli-parameters q_i te bepalen. We leiden limieten af voor $\boldsymbol{q}^*$ voor het geval de belasting van het systeem extreem laag of extreem hoog is. Daarnaast wordt een partiële oplossing voor het optimaliseringsprobleem gegeven. Deze oplossing geeft expliciet de waarde van bepaalde componenten van $\boldsymbol{q}^*$. De overige componenten van $\boldsymbol{q}^*$ kunnen worden berekend met behulp van het machtreeksalgoritme. Deze aanpak kan echter nogal tijdrovend zijn voor systemen met een groot aantal wachtrijen. Daarom wordt een eenvoudige benaderingsmethode voorgesteld, waarmee relatief snel bijna-optimale combinaties van de Bernoulli-parameters kunnen worden bepaald.

In hoofdstuk 4 wordt de invloed van het *routeringsmechanisme* op de prestatie van het systeem bestudeerd. In de meeste polling-modellen bezoekt de bediende de verschillende rijen in een vaste volgorde. Deze bezoekvolgorde wordt ook wel periodieke routering genoemd. In dit hoofdstuk echter bezoekt de bediende de rijen volgens een *toevalsafhankelijk* routeringsmechanisme. Om precies te zijn, na een bezoek van de bediende aan wachtrij i wordt rij j bezocht met kans $p_{i,j}$. Dit routeringsmechanisme wordt ook wel Markov-routering genoemd. Polling-modellen met Markov-routering zijn in het algemeen niet exact analyseerbaar. We laten zien hoe het machtreeksalgoritme kan worden gebruikt voor het ana-

lyseren van polling-modellen met Markov-routering. Numerieke experimenten met het machtreeksalgoritme suggereren dat de totale hoeveelheid werk in het systeem onder Markov-routering in het algemeen groter is dan onder periodieke routering. We tonen aan dat een soortgelijke dominantie niet algemeen geldig is voor alle individuele verwachte wachttijden. Kwalitatieve eigenschappen van optimale combinaties van routeringskansen worden bestudeerd. Numerieke experimenten met het machtreeksalgoritme wijzen uit dat bij relatief veel rijen de optimale routeringskansen 0 of 1 zijn. Ook blijken de optimale routeringsmatrices veelal van een speciale eenvoudig interpreteerbare structuur te zijn. Tenslotte worden enkele vuistregels gegeven voor het construeren van optimale routeringsmatrices.

In hoofdstuk 5 worden polling-modellen aan de orde gesteld waarin de bediende mag blijven stilstaan bij een wachtrij wanneer zich geen klanten in het systeem bevinden. Dit in tegenstelling tot de meeste polling-modellen, waarin de bediende op elk moment óf aan het bedienen is óf aan het omschakelen is van de ene naar de andere rij. Polling-modellen met een 'stilstaande' bediende zijn in het algemeen niet exact analyseerbaar. Bovendien valt deze klasse van polling-modellen buiten het algemene kader van hoofdstuk 2. We laten zien hoe dit soort modellen toch kan worden geanalyseerd met behulp van het machtreeksalgoritme. We onderzoeken de invoed van verschillende systeemparameters op de 'winst' die kan worden geboekt door de bediende te laten stoppen bij bepaalde rijen. Uit numerieke resultaten blijkt dat de prestatie van het systeem sterk verbeterd kan worden door de bediende te laten stoppen bij bepaalde wachtrijen, met name wanneer de systeembelasting vrij laag is en wanneer de omschakeltijden significant zijn. Daarnaast wordt ingegaan op het probleem van het bepalen bij *welke* wachtrijen de bediende het 'best' kan blijven stilstaan. In het algemeen is dit probleem niet oplosbaar. Daarom wordt een eenvoudige heuristische aanpak voorgesteld die in veel gevallen nauwkeurige resultaten oplevert.

In hoofdstuk 6 bestuderen we de prestatie van polling-modellen met *meerdere bedienden*, waarbij elk van de bedienden de rijen in een vaste volgorde bezoekt. Deze generalisatie van het klassieke polling-model met één bediende komt op natuurlijke wijze voor in veel praktische situaties. Echter, voor dit soort polling-modellen zijn in het algemeen maar weining structuurresultaten te verkrijgen. Er wordt aangetoond hoe polling-modellen met meerdere bedienden kunnen worden geanalyseerd met behulp van het machtreeksalgoritme. Daarmee kunnen de kansverdelingen van de rijlengten en van de posities van de bedienden in het systeem worden bepaald, evenals de verdelingen van de wachttijden van de klanten en de belasting die gedragen wordt door elk van de bedienden. Numerieke experimenten wijzen uit dat de bedienden de neiging vertonen bij elkaar 'in de buurt' te blijven, met name wanneer de bedienden de rijen in dezelfde volgorde bezoeken en het systeem zwaarbelast is. De resultaten geven ook aan dat dit clustereffect in het algemeen leidt tot langere gemiddelde wachttijden. Daarnaast wordt de prestatie van polling-modellen

met meerdere bedienden vergeleken met de prestatie van polling-modellen met één bediende met dezelfde belasting per bediende. De resultaten suggereren dat polling-modellen met meerdere bedienden in het algemeen beter presteren dan polling-modellen met één bediende. Tenslotte wordt een nieuwe benaderingsmethode voorgesteld waarmee de verwachte wachttijden snel en nauwkeurig bepaald kunnen worden.